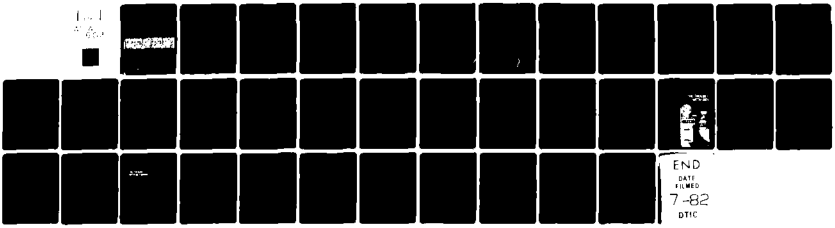
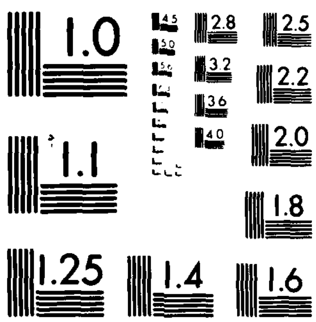
AD-A116 031 CALIFORNIA UNIV SAN DIEGO LA JOLLA CENTER FOR HUMAN --ETC F/G 5/8
FIVE PAPERS ON HUMAN-MACHINE INTERACTION.(U)
MAY 82 D A NORMAN N00014-79-C-0323
UNCLASSIFIED CHIP-112 ONR-8205 NL





MICROCOPY RESOLUTION TEST CHART

12

AD A116031

FIVE PAPERS ON HUMAN-MACHINE INTERACTION:

* **Some Observations on Mental Models**
* **A Psychologist Views Human Processing: Human Errors and Other Phenomena Suggest Processing Mechanisms**
* **Steps toward a Cognitive Engineering: Design Rules Based on Analyses of Human Error**
* **The Trouble with UNIX**
* **The Trouble with Networks**

Donald A. Norman

CENTER FOR HUMAN INFORMATION PROCESSING

The research reported here was conducted under Contract N00014-79-C-0323, NR 157-437 with the Personnel and Training Research Programs of the Office of Naval Research, and was sponsored by the Office of Naval Research and the Air Force Office of Scientific Research. The views and conclusions contained in this document are those of the author and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsoring agencies. Approved for public release; distribution unlimited. Reproduction in whole or in part is permitted for any purpose of the United States Government. ONR REPORT 8205

UNIVERSITY OF CALIFORNIA, SAN DIEGO
LA JOLLA, CALIFORNIA 92093

CHIP 112
MAY 1982

CHIP Report 112
May 1982

Five Papers on Human-Machine Interaction:

* Some Observations on Mental Models
* A Psychologist Views Human Processing: Human Errors and other Phenomena Suggest Processing Mechanisms
* Steps toward a Cognitive Engineering: Design Rules Based on Analyses of Human Error
* The Trouble with UNIX
* The Trouble with Networks

Donald A. Norman

Cognitive Science Laboratory

University of California, San Diego

Center for Human Information Processing

University of California, San Diego

La Jolla, California 92093

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ONR-8205	2. GOVT ACCESSION NO. AD-A116 031	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Five Papers on Human-Machine Interaction		5. TYPE OF REPORT & PERIOD COVERED Technical Report
		6. PERFORMING ORG. REPORT NUMBER Chip 112
7. AUTHOR(s) Donald A. Norman		8. CONTRACT OR GRANT NUMBER(s) N00014-79-C-0323
9. PERFORMING ORGANIZATION NAME AND ADDRESS Center for Human Information Processing University of California, San Diego La Jolla, California 92093		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS NR 157-437
11. CONTROLLING OFFICE NAME AND ADDRESS Personnel and Training Research Programs Office of Naval Research (Code 442) Arlington, Virginia 22217		12. REPORT DATE May 1982
		13. NUMBER OF PAGES 30
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

This research was also supported by the Air Force Office of Scientific Research.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Artificial Intelligence
Cognitive Engineering
Friendly Systems
Human-Computer Interaction
Mental Models

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

(OVER)

ABSTRACT

This report consists of five brief papers on different aspects of human-machine interaction. The first paper, "Some Observations on Mental Models," discusses the role of a person's mental model in the interaction with systems. The second paper, "A Psychologist Views Human Processing: Human Errors and other Phenomena Suggest Processing Mechanisms," discusses the differences between conventional digital processing structures (the Von Neumann machine) and the mechanism of the human. The third paper, "Steps toward a Cognitive Engineering," shows how analysis of error can lead to design principles. The fourth paper, "The Trouble with UNIX," is an informal critique of the UNIX operating systems. The final paper, "The Trouble with Networks," describes some of the computer interactions that resulted from the distribution of the fourth paper.

Some Observations on Mental Models

Donald A. Norman
University of California, San Diego

One function of this chapter is to belabor the obvious; people's views of the world, of themselves, of their own capabilities, and of the tasks that they are asked to perform, or topics they are asked to learn, depend heavily on the conceptualizations that they bring to the task. In interacting with the environment, with others, and with the artifacts of technology, people form internal, mental models of themselves and of the things with which they are interacting. These models provide predictive and explanatory power for understanding the interaction. These statements hardly need be said, for they are consistent with all that we have learned about cognitive processes and, within this book, represent the major underlying conceptual theme. Nonetheless, it does not hurt to repeat them and amplify them, for the scope of the implications of this view is larger than one might think.

In the consideration of mental models we need really consider four different things: the _target system_ the _conceptual model_ of that target system, the user's _mental model_ of the target system, and the _scientist's conceptualization_ of that mental model. The system that the person is learning or using is, by definition, the _target system_. _A conceptual model_ is invented to provide an appropriate representation of the target system, appropriate in the sense of being accurate, consistent, and complete. Conceptual models are invented by teachers, designers, scientists, and engineers.

To be published in D. Gentner and A. Stevens (Eds.), _Mental Models_. Hillsdale, N. J.: Erlbaum, 1982.

This research was conducted under Contract N00014-79-C-0323, NR 157-437 with the Personnel and Training Research Programs of the Office of Naval Research, and was sponsored by the Office of Naval Research and the Air Force Office of Scientific Research. I thank Sondra Buffett for her suggestions for the manuscript.

Mental models are naturally evolving models. That is, through interaction with a target system, people formulate mental models of that system. These models need not be technically accurate (and usually are not), but they must be functional. A person, through interaction with the system, will continue to modify the mental model in order to get the necessary results. Mental models will be constrained by such things as the user's technical background, previous experiences with similar systems, and the structure of the human information processing system. The *Scientist's conceptualization* of a mental model is, obviously, a model of a model.

Some Observations on Mental Models

My observations on a variety of tasks, with a wide variety of people, lead me to a few general observations about mental models:

1. Mental models are incomplete.

2. People's abilities to "run" their models are severely limited.

3. Mental models are unstable: people forget the details of the system they are using, especially when those details (or the whole system) have not been used for some period.

4. Mental models do not have firm boundaries: similar devices and operations get confused with one another.

5. Mental models are "unscientific": people maintain "superstitious" behavior patterns even when they know they are unneeded because they cost little in physical effort and save mental effort.

6. Mental models are parsimonious: often people do extra physical operations rather than the mental planning that would allow them to avoid those actions; they are willing to trade-off extra physical action for reduced mental complexity. This is especially true where the extra actions allow one simplified rule to apply to a variety of devices, thus minimizing the chances for confusions.

Let me now expand upon these remarks. In my studies of human error and human-machine interaction, I have made reasonably extensive observation of people's interactions with a number of technological devices. The situations that I have studied are quite diverse, including such tasks as the use of calculators, computers, computer text editors, digital watches and cameras, video cameras and recorders, and the piloting of aircraft. Some of these have been studied extensively (the computer text editor), others only in informal observation. I conclude that most people's understanding of the devices they interact with is suprisingly meager, imprecisely specified, and full of inconsistencies, gaps, and

idiosyncratic quirks. The models that people bring to bear on a task are not the precise, elegant models discussed so well in this book. Rather, they contain only partial descriptions of operations and huge areas of uncertainties. Moreover, people often feel uncertain of their own knowledge -- even when it is in fact complete and correct -- and their mental models include statements about the degree of certainty they feel for different aspects of their knowledge. Thus, a person's mental model can include knowledge or beliefs that are thought to be of doubtful validity. Some of this is characterized as "superstitious" -- rules that "seem to work," even if they make no sense. These doubts and superstitions govern behavior and enforce extra caution when performing operations. This is especially apt to be the case when a person has experience with a number of different systems, all very similar, but each with some slightly different set of operating principles.

Observations of Calculator Usage

Let me briefly review some of my observations on people's use of calculating machines. I observed people using hand-held versions of four-function, algebraic, and stack calculators while they were solving a series of arithmetic problems. They were asked to "think aloud" as they did the problems and I watched and recorded their words and actions. When all problems were complete, I questioned them about the methods they had used and about their understanding of the calculator.[1] Although the people I observed were all reasonably experienced with the machines on which I tested them, they seemed to have a distrust of the calculator or in their understanding of the details of calculator mechanics. As a result, they would take extra steps or decline to take advantage of some calculator features, even when they were fully aware of their existence. Most of the people I studied had experience with several different calculators, and as a result they mixed up the features. They were often unsure which feature applied to which calculator. They had various superstitions about the operations of the calculator. And finally, their estimation of the amount of mental workload required by various strategies often determined their actions; they would perform extra operations in order to reduce the amount of mental effort. Let me provide some examples.

---

1. The inspiration for these studies came from Richard Young's analyses of calculator operation, presented at the conference that led to this book. However, his work did not include any studies of what people actually believed of the calculators or how they used them: hence my investigations. I made up problems that required only simple arithmetic operations -- addition, subtraction, multiplication, and division -- but some required storage registers, writing down of partial results, or planning of the sequence to avoid the need for writing or storage.

Since performing these studies and writing the paper I have learned of the closely related observations and analyses made by Mayer and Bayman (1981).

One of the subjects I studied (on a four-function calculator) was quite cautious. Her mental model seemed to contain information about her own limitations and the classes of errors that she could make. She commented: "I always take extra steps. I never take short cuts." She was always careful to clear the calculator before starting each problem, hitting the clear button several times. She wrote down partial results even where they could have been stored in the machine memory. In a problem involving "constant sums," she would not use the calculator's memory because:

> I would not have done that because often when you play with the memory and the clear button, if you are not really clear about what it actually clears you can clear out the memory and it -- it -- I'm too cautious for that. I would be afraid that I'd mess up the memory.

All the people I observed had particular beliefs about their machines and about their own limitations, and as a result had developed behavior patterns that made them feel more secure in their actions, even if they knew that what they were doing was not always necessary. A major pattern that seemed to apply to all my calculator studies was the need for clearing the registers and displays. The four-function calculator did need to be cleared before starting new problems, but the stack and algebraic calculators did not. Yet, these people always cleared their calculators, regardless of the type. Moreover, they would hit the clear button several times saying such things as "you never know -- sometimes it doesn't register," or, explaining that "there are several registers that have to be cleared and sometimes the second and third clears do these other registers." (The four-function calculator that I studied does require two depressions of the CLEAR button to clear all registers.)

In an interesting complement to the excessive depressing of CLEAR to ensure that everything got cleared, during a problem with the four-function calculator where it became necessary to clear the display during the solution of a problem, one person balked at doing so, uncertain whether this would also clear the registers. All the people I observed expressed doubts about exactly what did and did not get cleared with each of the button presses or clear keys (one of the algebraic calculators has 3 different clear keys). They tended toward caution: excessively clearing when they wanted the calculator to be restarted, and exhibiting reluctance to use CLEAR during a problem for fear of clearing too much.

A similar pattern applied to the use of the ENTER button on the stack calculator. They would push it too much, often while commenting that they knew this to be excessive, but that is what they had learned to do. They explained their actions by saying such things as "It doesn't hurt to hit it extra" or "I always hit it twice when I have to enter a new phrase -- its just a superstition, but it makes me feel more comfortable."

These behaviors seem to reflect some of the properties of mental models, especially the ease of generating rules that have great precision and of keeping separate the rules for a number of very similar, but different devices. The rule to hit the CLEAR button excessively allows the user to avoid keeping an accurate count of the operation. Moreover, it provides a rule that is functional on all calculators, regardless of design, and that also makes the user resistant to slips of action caused by forgetting or interference from other activities. All in all, it seems a sensible simplification that eases and generalizes what would otherwise be a more complex, machine specific set of knowledge.

When people attribute their actions to superstition they appear to be making direct statements about limitations in their own mental models. The statement implies uncertainty as to mechanism, but experience with the actions and outcomes. Thus, in this context, superstitious behavior indicates that the person has encountered difficulties and believes that a particular sequence of actions will reduce or eliminate the difficulty.

Finally, there seemed to be a difference in the trade-off between calculator operations and mental operations that the people I studied were willing to employ. For problems of the sort that I was studying, the four-function machine was the most difficult to use. Considerable planning was necessary to ensure that the partial answers from the subparts of the problem could be stored in the machine memory (most four-function calculators only have one memory register). As a result, the users seemed to prefer to write down partial sums and to do simple computation in their heads rather than with the machine. With the stack machine, however, the situation is reversed. Although the machine is difficult to learn, once it is learned, expert users feel confident that they can do any problem without planning: They look at the problem and immediately start keying in the digits.

On Modeling a Mental Model

Consider the problem of modeling some particular person's mental model of some particular target system. Let the particular target system be called $\underline{t}$. Before we can understand how a person interacts with a target system, we need to have a good conceptualization of that system. In other words, we need a conceptual model of the system: call the conceptual model of $\underline{p}$, $\underline{C}(\underline{t})$. And now let the user's mental model of that target system be called $\underline{M}(\underline{t})$.

We must distinguish between our conceptualization of a mental model, $\underline{C}(\underline{M}(\underline{t}))$, and the actual mental model that we think a particular person might have, $\underline{M}(\underline{t})$. To figure out what models users actually have requires one to go to the users, to do psychological experimentation and

observation. [2]

In order to effectively carry out such observation and experimentation, we need to consider both representational and functional issues. Let me discuss three of the necessary properties: belief systems, observability, and predictive power.

These three functional considerations -- belief systems, observability, and predictive power -- apply to both the mental model and our conceptualization of the model, to both M(t) and C(M(t)). They can be summarized in this way:

Belief system. A person's mental model reflects the person's beliefs about the physical system, acquired either through observation, instruction, or inference. The conceptual model of the mental model C(M(t)), should contain a model of the relevant parts of the person's belief system.

Observability. There should be a correspondence between the parameters and states of the mental model that are accessible to the person and the aspects and states of the physical system that the person can observe. In the conceptual model of the mental model, this means that there should be a correspondence between parameters and observable states of C(M(t)) and the observable aspects and states of t.

Predictive power. The purpose of a mental model is to allow the person to understand and to anticipate the behavior of a physical system. This means that the model must have predictive power, either by applying rules of inference or by procedural derivation (in whatever manner these properties may be realized in a person); in other words, it should be possible for people to "run" their models mentally. This means that the conceptual mental model must also include a model of the

---

2. Let me warn the non-psychologists that discovering what a person's mental model is like is not easily accomplished. For example, you cannot simply go up to the person and ask. Verbal protocols taken while the person does a task will be informative, but incomplete. Moreover, they may yield erroneous information, for people may state (and actually believe) that they believe one thing, but act in quite a different manner. All of a person's belief structures are not available to inspection, especially when some of those beliefs may be of a procedural nature. And finally, there are problems with what is called the "demand structure" of the situation. If you ask people why or how they have done something, they are apt to feel compelled to give a reason, even if they did not have one prior to your question. They are apt to tell you what they believe you want to hear (using their mental models of your expectations). Having then generated a reason for you, they may then believe it themselves, even though it was generated on the spot to answer your question. On-line protocols generated while in the act of problem solving and that give descriptions of activities rather than explanatiors are much more reliable.

relevant human information processing and knowledge structures that make it possible for the person to use a mental model to predict and understand the physical system.

On the Relationship between Conceptual and Mental Models

Conceptual models are devised as tools for the understanding or teaching of physical systems. Mental models are what people really have in their heads and what guide their use of things. Ideally, there ought to be a direct and simple relationship between the conceptual and the mental model. All too often, however, this is not the case.

That a mental model reflects the user's beliefs about the physical system seems obvious and has already been discussed. What is not so obvious is the correspondence that should hold between the mental model and a conceptual model of the physical system, that is, between $M(t)$ and $C(t)$.

In the literature on mathematical learning models, Greeno and Steiner (1964) introduced the notion of "identifiability." That is, they pointed out that a useful model will have a correspondence between the parameters and states of the model and the operation of the target system. I find that these remarks apply equally well to the problems of mental models. It is important that there be a correspondence between the parameters and states of one's model and the things one is attempting to describe. This restriction does pose some strong constraints upon the nature of the mental model. Certain kinds of mental models will be ruled out if the identification cannot be easily made.

A major purpose of a mental model is to enable a user to predict the operation of a target system. As a result, the predictive power of such a model is of considerable concern. Although great stress is laid in this book to the notion of "running" a conceptual or mental model, it should also be possible to make predictions by straightforward inference, a declarative form of predictability, rather than the implied notion of procedural running of a model. Whatever the mechanism, it is clear that prediction is one of the major aspects of one's mental models, and this must be captured in any description of them.

The System Image

In the ideal world, when a system is constructed, the design will be based around a conceptual model. This conceptual model should govern the entire human interface with the system, so that the image of that system seen by the user is consistent, cohesive, and intelligible. I call this image the <u>system image</u> to distinguish it from the conceptual model upon which it is based and the mental model one hopes the user will form of the system. The instruction manuals and all operation and teaching of the system should then be consistent with this system image. Thus, the instructors of the system would teach the underlying conceptual model to the user and, if the system image is consistent with that model, the user's mental model will also be consistent.

For this to happen, the conceptual model that is taught to the user must fulfill three criteria:

Learnability
Functionality
Usability

What good is a conceptual model that is too difficult to learn? Or a model that has little functionality, failing to correspond to the system image or failing to predict or explain the important aspects of the target system? Or what of a conceptual model that cannot easily be used, given the properties of the human information processing structure with its limited short term memory and limited ability to do computations?

Alas, all too often there is no correspondence among the conceptual model of the system that guided the designer, the system image that is presented to the user, the material in the instructional manuals and that is taught to the user, and the mental models of the user. Indeed, for many target systems, there is no single conceptual model that was followed in the design. The stack calculator gives us a good positive instance where a conceptual design was neatly implemented into a consistent physical device, with the operations and instructions all based around the same basic model. It should be no surprise, therefore, that in my studies, users of this calculator were most confident of their abilities.

References

Greeno, J. G., & Steiner, T. E. Markovian processes with identifiable states: General considerations and applications to all-or-none learning. Psychometrika, 1964, 29, 309-333. (An easier treatment is provided in the chapter on identifiability in Restle, F., & Greeno, J. G., Introduction to mathematical psychology. Reading, Mass.: Addison-Wesley, 1970.)

Mayer, R. E., & Bayman, P. Psychology of calculator languages: A framework for describing differences in users' knowledge. Communications of the ACM, 1981, 24, 511 - 520.

A PSYCHOLOGIST VIEWS HUMAN PROCESSING: HUMAN ERRORS AND OTHER PHENOMENA SUGGEST PROCESSING MECHANISMS

Donald A. Norman

Program in Cognitive Science
University of California, San Diego
La Jolla, California 92093

ABSTRACT

I argue from studies of human performance, including slips of action and skilled typing that human processing structures are of a special sort, with weak binding between functions and arguments, with strong excitatory and inhibitory interactions among simultaneous processes, and with the parts of action sequences neither strongly ordered nor tightly coupled. I argue that analyses of human performance imply a class of processing structures quite different than is commonly envisioned within Artificial Intelligence.

I. How do people do more than one thing at a time?

An important aspect of everyday behavior is that we do several different activities at the same time, oftentimes for simultaneous (and possibly conflicting) purposes. Even when we do not attempt simultaneous actions, we still might be planning or reviewing one set of things while performing or accomplishing another. We delay and defer goals or actions as needed, waiting for appropriate times for them to be accomplished. This occurs for several reasons. Some biological goals do not need to be satisfied at any particular instant, but within reason, can be executed at convenience (e.g., such things as eating, sleeping, or toilet activities). Some daily tasks have similar characteristics (e.g., going to the bank or post office, purchasing some needed item). Some tasks have to be deferred because there is not sufficient time or information to complete them during one session of work (e.g., writing a scientific paper, reading a book, learning a complex task). Finally, even for tasks that are continually active from start to completion, they may span such a long duration that other things are also done along the way, and the individual components of the major task may have to wait for minutes or even hours before being executed.

These problems appear to be analogous to the scheduling problems of modern real-time computers, and some of the analyses from that field are relevant. However, the human is a special kind of biological processor, and I suspect that suprisingly little of what we know of time-shared computers applies to the human. The difficulties in doing two or more tasks at the same time are well known. There are only two ways that a system can do two or more things together at the same time. One way is to have sufficient processing machinery that the two tasks use different resources and do not interact. The second way is to switch back and forth between the two, saving the complete status of the current state before switching tasks and then restoring the state completely when switching back.

Which method do people use? There is clear evidence for both. Different processing structures control walking and talking, eating and seeing. The same processors are switched among tasks when we listen first to this conversation, then to that, or when driving a car while conversing, looking first at the road, then at the passenger. The interesting cases arise when neither solution seems applicable because the several tasks interact with one another. Some kinds of mental activity can cause pupil size to increase (as expert poker players know). We hear better in the direction in which we are looking (even if we do not turn the head) and, conversely, it is hard to ignore the sounds from the direction in which we look, even when we are trying to listen to something else [12]. Novices cannot tap different rhythms with different hands, but expert musicians can. Thoughts can intrude upon actions. The initiation of action can interrupt thought. Emotions -- and even such things as hunger -- can disrupt thought. We tend to remember sad things when we are sad and happy things when we are happy. Different parts of our system intrude upon others, apparently in subtle, continuous ways, a point exploited by Freud.

A. Studies of attention (psychology) and time-sharing (computer science) do not provide helpful information

These characteristics of real behavior pose some interesting and important puzzles for students of human information processing. We know little about how such multiple goals and tasks get scheduled and accomplished. There has not been much study of this aspect of behavior. This is especially true in view of the psychological literature on simultaneous attention that argues strongly for limits on our ability to do several tasks at any one time. I myself have argued such a point, telling audiences of undergraduates how people are limited to doing roughly one thing at a time. Of course, while I say this, I am pacing back and forth in front of the class, avoiding the table and chair in my path, juggling a piece of chalk from hand to hand, planning the remainder of the lecture, and worrying about how I am going to get through the demands of the rest of the day. My actions contradict my speech, but in actuality, it is even worse. Any one of the "single" things I am doing is a complex set of overlapping activities. The act of speaking, for example, involves many components, many of which should really be considered separate tasks. In speaking, there is the high level planning of the utterance, the formation of the structure of the sentences, the proper morphological selection and construction of the words, and the complex control of the speech organs and of the numerous muscles in the face, mouth, throat, and chest that must operate in parallel with overlapping control signals. Thus, even a so-called single task is really many simultaneous tasks.

Psychologists deal with this apparent contradiction between theoretical belief and reality by talking of the distinction between automatic and non-automatic actions, stating that automatic acts are not under conscious control and do not require attentional resources. As a result, there is no limit on how many of these can be done at any one time, as long as there is no conflict in the use of any particular physical or psychological structure. The trouble with this explanation is that it doesn't tell us anything about how it is actually accomplished. To be polite to my field, I will make the

....

Research support was provided by the Office of Naval Research and the Naval Air Development Center under contract N00014-79-C-0323

Published in the Proceedings of the International Joint Conference on Artificial Intelligence, Vancouver, 1981.

excuse that this explanation is still at an infant stage of development. The statement allows us to reconcile our observations of real behavior with the theoretical belief in attentional limitations by saying that, well, not everything requires attention, not once it is well learned. Regardless of what you might think of the statement and of the lack of specificity in the arguments, there is another problem with the approach; the attentional limitations are only part of the problem. We still don't know how any organism can simultaneously perform many tasks. What kind of structures are necessary, and which are really present in the human? How can we account for the errors that people make?

Now turn to computers and, in particular, the work in Artificial Intelligence. Let me quickly assure half of my audience -- and warn the other half -- that AI doesn't have any idea of how to handle the problem either. The relationship between the study of Artificial Intelligence and human intelligence should go two ways, and although psychologists have often taken more from AI than AI has taken from us, I think the direction of the information flow is changing. In this case we are equals; there is only a weak ebb between two stagnant pools.

Most of the intelligent programs that have so far been developed within AI are single minded, experts in their single domain of inquiry, but unable to deal with any other domain. Even when there are systems that can deal with several different domains or sub-domains of a topic, they do them in a sensible fashion, one at a time, rather than in the inelegant, cluttered human fashion of attempting to think of everything at once, mixing up the concepts of one with those of the other. The virtue for the computer is elegance and power. The virtue for the human is creativity and flexibility.

B. *Human error is suggestive of a special form of mechanism*

I want to argue for a different kind of processing mechanism than is usually considered by people within Artificial Intelligence. In the end, it may not be wise to model many aspects of human intelligence with conventional processing structures. But before I get to that, let us review the argument.

The multiple-purpose, multiple processing aspect of our behavior leads to difficulties. I have already listed some phenomena that imply interactions among processing structures. In this paper I concentrate upon the form of human errors. Thus, we make errors. We are easily distracted by events, stopping to do things we had not intended, or we are captured by habitual acts, performing them instead of the ones intended. At times, we can be data driven, responding to sensory signals, whether we intended to or not. This can be useful, for it allows us to react appropriately to unexpected events in the environment. It is not so useful when data driven processing interrupts our intended actions, at times so distracting us from our intentions that we neglect to complete them. These errors imply that we neither separate the tasks well nor switch completely among them. As a result, we intermix components, lose track of our status on any given task, and oftentimes do the right thing on the wrong occasion.

Errors give insight into the system, for they offer powerful clues as to the operation of the underlying mechanism. We need not agree with Freud's view that "the meaning in them is unmistakable, even to the dullest intelligence, and strong enough to impress even the most critical judgment", but we can still agree that they are strongly suggestive. Errors can be divided into several different categories. I divide errors into two major classes: mistakes and slips, with the division being whether the error occurred prior to or after the formation of the highest level intention. I define a mistake to be an error in forming the intention. Thus, a mistake can result from knowledge that is erroneous or incomplete, either in the information that the person brings to the situation or that is available from the environment. The mistake can also arise in the psychological mechanisms of decision and planning that are involved in the formation of the intention. I define a slip to be a failure in carrying out the intention properly. That is, the appropriate action is started, but somewhere along the path of execution it is diverted or deflected.

There are several collections of slips [2,3,9,11]. The instances are both humorous and informative. A business executive roared "Come in" instead of "Hello" when answering the telephone. A friend politely said "Come in" instead of "Sit down" when inviting a new person to join the two of us at a table in a hotel restaurant.* Pilots have raised the landing gear instead of the flaps. One person reported cleaning a fish and throwing the cleaned fish overboard, keeping the entrails. In preparing for a party, one person put the cake in the refrigerator and the salad in the oven. Computer users report numerous errors: typing commands into the text editor while in "insert" mode, or text while in "command" mode; deleting files instead of moving them. There are data-driven errors, in which the sight of something leads to an unintended action -- one of my students calls this the "parking spot error": if you come across a parking spot while driving through a city, you may find yourself parked in it, even if you had no intention of stopping there. (The same student reports dashing into an elevator that happened to open its doors just as he was walking by, even though he hadn't meant to take an elevator.) A reasonably common typing error is the "doubling error": doubling the wrong letter in a word, yielding **bokk** or **claas** instead of **book** or **class**.

Examples of slips can be found in both speech and motor actions. One example is to select the wrong word, as in: "Wouldn't it be cheaper, I mean faster, to go that way?" I classify this as a "description" error, one that results from an error in memory retrieval. The word that was first retrieved shares features of the semantic description of the intended word. The error can perseverate, as in: "They have Chinee -- Japa -- Mexican food to go."

There are other forms of verbal slips. A blend occurs when two competing patterns are merged, as in "clut" which merges "close" and "shut." In a Spoonerism, components of the words are interchanged, as in: "Ruman and Normalhart" instead of the intended "Norman and Rumelhart." (The examples come from Norman [9], and Fromkin [2,3].)

Freud made the point that most errors have multiple causes, and that seems to be true of these as well. For the several people who have reported going to their bedroom to change clothes for dinner and finding themselves undressed and in bed, they may have been "captured" by performing the initial stages of a familiar habit and unconsciously completing the familiar instead of the intended, but they may also have been unconsciously attempting to avoid the dinner. The invitation to "Come in" to the restaurant table could have been affected by the fact that one of us was sitting in a semi-enclosed booth (and the person who made the slip so argued). In my experience, these subtle, clinical interpretations seem initially far-fetched, but are confirmed with surprising frequency by the people who make the slips. Thus the puzzle for those who wish to figure out the mechanism; how do different sources of information interact to lead to slips (while also accounting for the fact that most of our actions are correct)?

The various phenomena I have described, plus others, imply that the parts of action sequences are neither strongly ordered nor tightly coupled. That is, I think that the biological system is struc-

* All the slips reported in this paper have been collected with some care as to accuracy, and with the original intention verified by the perpetrator. See the original publications for details.

tured so as to use ambiguous information for memory search, to allow itself to be responsive to multiple sources of information, to combine and overlap data paths, and to deliberately intermix what one would have thought to be independent processing streams. Although these properties can lead to errors, I believe that they are also exactly the sort of thing that gives us much of the power of human creativity and judgement, to allow us to be tolerant of noise and of error, to behave flexibly, to respond in imaginative and creative ways to novel events, and to be able to shift our strategies and behavior when the situation shifts.

The basic concept is simple. We assume that the human information processing system is mediated by means of many separate processing structures, each of which can do only simple operations, but each of which is coupled to numerous other structures. We call these structures *schemas*, and we allow each to have an activation value that excites or inhibits its neighboring schemas and is triggered into controlling an action sequence whenever the combination of its activation value and the goodness-of-fit of its specific trigger conditions exceed a threshold value. (For a closely related argument and description of computational structures, see [1].) For present purposes, *all that is needed is the understanding* that there are independent processing structures, each capable of controlling action, and that synchronization and cooperation among them is handled by activation and inhibition links among schemas. More discussion can be found in [7,8,9,10,13].

TABLE 1
CLASSIFICATION OF ACTION SLIPS
(Adapted from Norman, [9])

- I. Slips in the formation of the intention
  - A. Mode errors: erroneous classification of the situation
  - B. Description errors: ambiguous or or incomplete specification of the intention
- II: Slips that result from faulty activation of schemas
  - A. Unintentional activation
    - 1. Capture errors: when the intended sequence is similar to another, better learned or more frequent sequence, the latter may gain control
    - 2. Data-driven activation: external events activate schemas
    - 3. Associative activation: currently active schemas activate others with which they are associated
  - B. Loss of activation
    - 1 Forgetting an intention (but continuing with the sequence)
    - 2 Misordering the components of a sequence
    - 3 Leaving out steps in a sequence
    - 4 Repeating steps in a sequence
- III: Slips that result from faulty triggering of active schemas
  - A. False triggering a properly activated schema is triggered at an inappropriate time
    - 1. Spoonerisms: reversals of event components
    - 2 Blends combinations of components from two competing schemas
    - 3 Thoughts leading to actions triggering of schemas only meant to be thought, not executed
    - 4 Premature triggering
  - B Failure to trigger
    - 1 The action was preempted by competing schemas
    - 2 There was insufficient activation
    - 3 The trigger conditions failed to match

Action slips come in many different varieties. I have attempted the analysis shown in Table 1, based upon a theoretical framework that assumes that actions are caused by the activation and triggering of schemas.

II. Studies of skilled behavior provide more clues

A. Skilled typing has interesting properties

Another source of information about how people do simultaneous actions comes from the study of skilled tasks. One such task is expert typing, and detailed study of the typist reveals some interesting insights into the nature of skilled human performance. Typing is a single task that requires multiple control of the 10 fingers and 2 hands -- there are 60 tendons and 30 joints involved simply in the movement of the fingers. Study of typing is today one of the major themes in our laboratory, and the analyses of typing errors and typing performance tell us quite a bit about the nature of cooperative interaction among simultaneous activities. At this point I will only mention two aspects of skilled typing. One is the doubling error in which the wrong letter in a word is doubled, so that a word like **book** or **manner** is typed as **bokk** or **maaner**. The other is the overlapping nature of the execution of the finger movements [4]. The finger movements start several letters ahead of their scheduled arrival time, oftentimes out of sequence of the final temporal order in which they are made. It is as if each finger starts as soon as it can towards its intended target, and the hand appears to cooperate, configuring itself so as to make maximum movement towards as many targets at a time as possible.

This latter example is important, for it illustrates a situation in which simultaneous tasks cooperate rather than compete. This *cooperation among possible competitive tasks happens frequently.* Suppose you wish to pick up several pencils and a piece of paper at the same time, using only one hand. The normal finger movements that would be performed were only one object to be picked up are modified to allow for cooperation among the fingers and hand to accomplish the multiple goal. I predict that one of the changes that occur in performance as a person becomes expert is a change from mutual competition of simultaneous actions to mutual cooperation. The behavior therefore changes from doing but a single action at a time to overlapping, cooperative performance of several simultaneous acts.

TABLE 2
THE BASIC PHENOMENA OF TYPING
(Adapted from Rumelhart & Norman, [14])

- I. The timing of keystrokes
  - A. People can type very quickly.
  - B. Cross hand interstroke intervals are shorter than those within hands.
  - C. Within hand interstroke intervals appear to be a function of the reach from one to the next.
  - D. The time for a particular interstroke interval can depend on the context in which it occurs.
  - E. There is a negative correlation between the intervals on successive strokes--especially when the alternate strokes occur on alternate hands.
- II. Pattern of Errors
  - A. Transposition Errors
  - B. *Doubling Error*
  - C. Alternation reversal errors
  - D. Homologous errors
  - E. Capture errors
  - F. Omission errors
  - G. Misstrokes
- III. The general organization of typing
  - A. Skilled typists move their hands towards the keys in parallel
  - B The units of typing seem to be largely at the word level or smaller
  - C Sequences involving cross hand strokes seem to take longer to program than those involving only within hand strokes

Studies of typing reveal a number of phenomena that provide considerable constraints on the possible mechanisms that could be responsible for the actions. A list of the phenomena we have examined is presented in Table 2.

B. The doubling error implies that there is no type-token distinction

Consider the doubling error. How could it come about? In our attempt to devise a formal model of the typing process [14], we took special note of errors of doubling and alternation. (An alternation occurs in a word like **these** in which the **e** alternates, but when typed, the wrong letter is alternated, as in **thses**.) The ex-

istence of of doubling and alternation errors pose special problems. Consider the word **book**. According to our arguments, the word would be represented by schemas for each of the letters: **b o o k**. It is easy to see how such a representation could lead to transposition errors (such as **boko**) but not to doubling errors. It would be easy to make up a schema for a doubled letter (so that the word would be represented by the schemas **b double-o k**), but this would not lead to the doubling errors either.

The doubling error turns out to have two major implications. First, it implies that there are special schemas that signal the existence of doubled letters, and that occasionally these schemas get applied to the wrong letters. In a computational terms, this means that the binding between the arguments of the special schemas for doubling occasionally get made improperly. Second, the need for a special schema to mark doubled letters implies a difficulty in having the regular letter schema signal the double. Why isn't the word book represented by the schemas **b o o k** ? The reason would seem to be that this would require two instances (types) of the schema for o; the existence of the doubling error implies that such repeated tokens of a schema might not be possible.

Thus, the existence of doubling errors forced us to a pure "type" model, in which each letter could only have a single keypress schema; the keypress schemas exist only as "types," with no "token" schemas. There must be a special schema that signals the presence of a doubled letter. Moreover, there must be a weak binding between the special schema and the arguments upon which it operates. In our model, we let the binding be established via activation values, with noise sometimes leading to errors in the binding. The existence of alternation errors led to the same conclusion; special schemas that signal the presence of alternating letters, with a weak binding between the schema and its arguments.

III. On possible psychological mechanisms

We see that there are several different aspects of skilled behavior:

1. Competition among actions, so that the doing of one thing inhibits the doing of another. For some combinations of actions, the mechanisms required are incompatible, so that the competition is necessary and in these cases some sort of priority or inhibitory processes are required.
2. Cooperation among actions, so that the operations of one action are modified to accommodate another. In this situation, most noticeable with skilled performers or with highly distinct, compatible actions, the simultaneous actions must engage in some process of "negotiation" to permit mutual performance. Thus, if one wishes to carry several objects at the same time -- for example, several pencils, a piece of paper, and a cup -- the normal movements and positions of the fingers, hands, and arms will be altered to make the cooperation possible.
3. Slips of performance, so that the components of one action sequence may get mixed up with the components of another, or the memory or the resource requirements for one will interfere with the requirements for the other, and so on.
4. Non-independence of action, so that the performance of one activity either affects or causes the performance of others, even when these other activities would appear to be quite unrelated. It is as if there were an overflow from the activation of one set of processing structures to neighboring structures, in which the major source of interaction results from physical proximity of the processing structures rather than from logical relationships among the activities being performed. (For an interesting review of this concept, see [6].)

These different aspects of simultaneous performance provide hints as to the nature of the underlying mechanisms. I have already suggested that the doubling error in typing says something of the underlying representational structure, and of the possible mechanisms for binding a function to its arguments. Slips provide constraints on the nature of the underlying representational and processing mechanism. Examination of skilled typing provides another source of evidence, requiring some mechanism that can yield cooperative behavior among the fingers and hands. Studies of attention and of neurological deficits provide yet another source of information.

In our attempt to construct a processing model of these aspects of human behavior we have been forced to deviate from the more traditional processing structures. Instead, we find that a viable structure seems to require multiple, parallel units, all interacting with one another, activating (and inhibiting) one another, with a tradeoff between activation value and the goodness to fit to trigger conditions. The scheme that we propose is a relative of production systems, but the control structure that we propose is somewhat different.

A. The role of will in the control of action

We postulate that skilled action sequences are automatic; no conscious control of them is necessary. However, because people sometimes perform an action when the conditions are not completely satisfactory, or hold back an action even when it would otherwise be appropriate some other form of control is required. In [10], we suggest that the normal configuration of schemas that perform an operation can be thought of as a *horizontal thread* of control (the name taken from the fact that the processing structure for some even sequence is often depicted as a series of horizontal processing stages). In normal circumstances, the horizontal thread suffices to carry out the action, with component schemas being triggered when their activation values and trigger conditions are satisfactory. However, attentional (conscious) control is necessary when there is concern about the adequacy of the horizontal thread structures (as in ill-learned tasks, novel situations, or situations perceived to be dangerous). This is done through control of the activation values of schemas by means of *vertical thread* structures. The application of attentional activation to bias the control of the horizontal thread schemas we called "will." Thus, by the exertion of will, one can cause a schema to be triggered even if it would otherwise not have been or to prevent a schema from being triggered that would otherwise have been.

The application of vertical thread activation, will, is best illustrated by the situation where one wishes to perform an undesirable act (such as getting out of bed on a cold morning) or to prevent a desirable act (such as eating any more of a rich and tasty desert). In both cases will is required, in the former to increase the activation values sufficiently to cause triggering of the schema even in the absence of a sufficiently good fit of the triggering conditions, and in the latter case, to prevent an activity, even though the normal activation values and triggering conditions have been met. In the latter case, continual attentional effort is required, for if attention lapses, the schema will revert to its normal activation values and triggering conditions, and the action will be performed.

The models of human processing suggested here need not be the only candidates. I mention them because they are suggestive of the sort of processing structures required to account for human performance. The important point is that conventional processing structures can not describe human behavior; a new breed of computational mechanism must be developed.

ACKNOWLEDGEMENTS

The form of processing structure described here was developed with my continual collaborator David Rumelhart, and owes a lot to the developments of Jay McClelland (as in [7,8,13,]). These in turn have been influenced by the work of Geoff Hinton, see [5] and Hinton, this conference.

REFERENCES

[1] Feldman, J. A. & Ballard, D. H. *Computing with connections.* Rochester, Department of Computer Science, University of Rochester Technical Report TR72. April, 1981.

[2] Fromkin, V. (Ed.). *Speech errors as linguistic evidence.* The Hague: Mouton, 1973.

[3] Fromkin, V. (Ed.). *Errors of linguistic performance: Slips of the tongue, ear, pen, and hands.* New York: Academic Press, 1980.

[4] Gentner, D. R., Grudin, J. & Conway, E. *Finger movements in transcription typing.* San Diego, California: Center for Human Information Processing, University of California, San Diego. Technical Report 8001. May, 1980.

[5] Hinton, G. E. & Anderson, J. A. (Eds.). *Parallel models of associative memory.* Hillsdale, N. J.: Lawrence Erlbaum Associates, 1981.

[6] Kinsbourne, M. & Hicks, R. "Functional cerebral space: A model for overflow, transfer, and interference effects in human performance." In J. Requin (Ed.), *Attention and Performance VII.* Hillsdale, N. J.: Lawrence Erlbaum Associates, 1978.

[7] McClelland, J. L. "Neural Models of Perception and Memory." *Scientific American*, in press.

[8] McClelland, J. L. & Rumelhart, D. E. "*An interactive activation model of the effect of context in perception, part I.*" *Psychological Review*, in press.

[9] Norman, D. A. "Categorization of action slips." *Psychological Review*, 1981, *88*, 1-15.

[10] Norman, D. A. & Shallice, T. *Attention to action: Willed and automatic control of behavior.* La Jolla, California: Center for Human Information Processing, University of California, San Diego, Technical Report 99, Dec. 1980.

[11] Reason, J. T. "Actions not as planned." In G. Underwood & R. Stevens (Eds.), *Aspects of consciousness.* London: Academic Press, 1979.

[12] Reisberg, D., Scheiber, R. & Potemken, L. "Eye position and the control of auditory attention." *Journal of Experimental Psychology: Human Perception and Performance.* 1981, 7, 318-323.

[13] Rumelhart, D. E. & McClelland, J. L. "An interactive activation model of the effect of context in perception, part II." *Psychological Review*, in press.

[14] Rumelhart, D. E. & Norman, D. A. *Simulating a skilled typist: A study of skilled cognitive-motor performance.* La Jolla, California: Center for Human Information Processing, University of California, San Diego. Technical Report 102. May 1981.

Steps toward a Cognitive Engineering:
Design Rules Based on Analyses of Human Error

Donald A. Norman

Program in Cognitive Science
University of California, San Diego
La Jolla, California 92093

Abstract

This paper uses the analysis of human error to provide a tool for the development of principles of system design, both to minimize the occurrence of error and to minimize its effects. Eventually, it should be possible to establish a systematic set of guidelines, with explicit, quantitative cost-benefit tradeoffs that can lead toward a design discipline -- a "Cognitive Engineering." This short note starts the process.

My eventual goal is the establishment of a discipline of "Cognitive Engineering" that can provide designers with the tools required to make their products more sensitive and responsive to the needs of the users. These tools will have at least two components: *first, a set of well established* procedures and methods with known benefits and costs, advantages *and disadvantages; second, a set* of quantitative modeling aids that can be used to give numerical assessment of the performance to be expected from a particular design choice. My hope is that by providing a rationale based upon modern cognitive theory, it will be possible to generalize these findings to new situations and to present them in such a way that designers will find them accessible and useable during the the course of design. This paper is simply the very beginning of the endeavor. There are several ways to begin. Let me describe three:

1. Start with the psychological mechanisms that have been studied by psychologists; use knowledge of the processing mechanisms to derive the important constraints on human performance. This is the approach taken by Card, Moran, and Newell (1982).

2. People form mental models of each other, the world, and of the devices and systems with which they interact. These mental models are used to predict system behavior and guide actions. The models, however, have interesting properties, sometimes being derived from idiosyncratic interpretations of the system. Moreover, the models must operate within the constraints of the human processing system (I expand on this in Norman, 1982). The study of these models, their veracity, and the capability of people to use the models wisely provides an important tool for the understanding of the human-system interface. This is the approach described in <u>Mental Models</u> (Gentner & Stevens, 1982).

3. Use analyses of people's performance in a variety of situations -- but especially their errors -- to construct an analysis of the appropriate form of human-machine interface that would optimize performance and minimize either the incidence of error or the effect of the error, once committed.

All three of these approaches are complementary to one another and should be combined in any complete attempt to produce a cognitive engineering. In this brief paper I use only the third approach.

<u>System Design Principles Can Be Derived from the Classes of Human Error</u>

I have collected a number of errors made by people, both in their everyday life and also in their use of computer systems (Norman, 1981). These errors yield some insight into the psychological mechanisms that are involved, but they can also be used to examine the human-machine interface.

Call the highest level specification of a desired action an <u>intention</u>. The intention may result from conscious decision making or from subconscious processing. The important point is that it is a high level specification that starts a chain of processing that normally results in the accomplishment of that intention. An error in the intention is called a "mistake." An error in carrying out the intention is called a "slip."

Slips can be classified into a small set of classes based upon the mechanisms that seem the most likely causes. The basic classification is based upon a simple model of the human in which it is assumed that any intention sets loose a number

---

Research support was provided by the Office of Naval Research, the Air Force Office of Scientific Research, and the Naval Air Development Center under contract N00014-79-C-0323. Requests for reprints should be sent to Donald A. Norman, Program in Cognitive Science C-015; University of California, San Diego; La Jolla, California, 92093, USA.

Published in the Proceedings of the Conference on Human Factors in Computer Systems, March 15-17, 1982, Gaithersburg, Maryland.

of active schemas, each with an activation value and a set of trigger conditions; a schema is performed whenever the combination of its activation value and the goodness of match of its trigger conditions reaches an appropriate level. This model gives rise to the classification of slips listed in Table 1. For current purposes it is only necessary to examine a subset of the classification scheme shown in Table 1. In particular, I discuss mode errors, description errors, capture errors, and activation errors.

Table 1
(modified from Norman, 1981).

A Classification of Slips Based on Their Presumed Sources

Slips in the formation of intention
- Mode errors: erroneous classification of the situation
- Description errors: ambiguous or incomplete specification of the intention

Slips resulting from faulty activation of schemas
- Unintentional activation: when schemas not part of a current action sequence become activated for extraneous reasons, then become triggered and lead to slips
  - Capture errors: when a sequence being performed is similar to another more frequent or better learned sequence, the latter may capture control
  - Data-driven activation: external events cause activation of schemas
  - Associative activation: currently active schemas activate others with which they are associated
- Loss of activation: when schemas that have been activated lose activation, thereby losing effectiveness to control behavior
  - Forgetting an intention (but continuing with the action sequence)
  - Misordering the components of an action sequence, including skipping steps and repeating steps

Slips resulting from faulty triggering of schemas
- False triggering: a properly activated schema is triggered at an inappropriate time
  - Spoonerisms: reversal of event components
  - Blends: combinations of components from two competing schemas
  - Thoughts leading to actions: triggering of schemas meant only to be thought, not to govern action
  - Premature triggering
- Failure in triggering: when an active schema never gets invoked because:
  - Action was preempted by competing schemas;
  - There was insufficient activation, either as a result of forgetting or because the initial level was too low;
  - There was a failure of the trigger condition to match, either because the triggering conditions were badly specified or the match between occurring conditions and the required conditions were never sufficiently close.

Mode Errors Suggest the Need for Better Feedback

Mode errors occur when the person believes the system is in one state (mode), whereas it is actually in another. This leads to the performance of an inappropriate action. Mode errors occur frequently in systems that do not provide clear feedback as to their current state. The most common examples in my collection come from the use of computer text editors, where users try to issue commands while in text mode or to type text while in command mode. Similar errors occur in pushing the buttons on complex digital watches. The autopilots of commercial aircraft provide numerous possibilities for mode errors; a recent incident in which an Aero Mexico DC-10 stalled, was badly buffeted, and lost the tips of both elevators appears to have been the result of a mode error in using the autopilot on the part of the crew (NTSB, 1980a). The clear implication of mode errors is that they result from inadequate feedback and indication of the state of the system.

Description Errors Suggest the Need for Better System Configuration

A description error occurs when there is insufficient specification of the action, and the resultant ambiguity leads to an erroneous act being performed. Usually this erroneous act is closely related to the desired one. Often the errors are humorous (at least to others). One dramatic case in my collection occurred when a person, while cleaning a fish in a rowboat in the middle of a lake, threw the cleaned fish overboard and kept the entrails. In another related case, a person preparing for a party put the cake in the refrigerator and the salad in the oven. Description errors also occur in operational situations, where they can lead to serious accidents. The two preceding description errors can be thought of as situations in which the proper arguments and functions were specified, but the ordering of the arguments was improper. In general, these errors occur when different actions have similar descriptions, either in the specification of the actions or in the class of arguments.

One class of description errors occurs in the use of computer text editors which have multiple commands, usually based upon one or two keystrokes. Thus, in the text editor "vi" (the screen editor supplied with the Berkeley Distribution of the UNIX operating system), each of the letters *d*, *f*, *g*, and *u* has different meanings when typed in lower case ("d"), upper case ("shift-d" or "D"), or as a control key ("control-d"). Many other keys also have these multiple uses. It should come as no surprise to discover that description errors occur frequently in this editor.

Description errors are relatively common in the throwing of switches or operations of controls, especially when the operations are similar, such as in the setting of altimeters, radio frequencies, and transponder codes. This problem is

especially bad in the design of nuclear power plant control rooms, where switches and controls are laid out in neat, logical, nice looking rows. The result, however, is clear potential for confusion, for reading the wrong instruments and for operating the wrong controls (Lockheed Missiles & Space Company, 1976).

Description errors can be expected to occur wherever control panels are designed so that at a quick glance (or in peripheral vision) the distinctions among controls are not clear enough. Solutions to this problem have long been known. Three principles are as follows:

1. Arrange instruments and controls in functional patterns, perhaps in the form of a flow chart of the system;

2. Use "shape coding" to make the controls and instruments look and feel different from one another;

3. Make it difficult to do actions that can lead to operations that have serious implications and that are not reversible.

With computer systems, these three principles are readily modified to their Computer ("C") versions:

1C. Screen displays and menu systems should be organized functionally;

2C. Design the command language (or menu display headings) to be distinct from one another so as not to be easily confused, either in perception or in the action required;

3C. Make it difficult to do actions that can lead to operations with serious implications and that are not reversible

<u>Lack of consistency in command structure leads to description errors</u>. One class of description errors occurs when a person attempts to re-derive an action sequence and does so improperly, forming a sequence appropriate for an action different from the one intended. This occurs primarily through a lack of consistency in command structure, so that the appropriate structure for one command is not the same for another, even though the commands appear to be related and share a common description of purpose, action, and even part of the command format. Similar situations occur in the interpretation of instrument readings. The basic concept involved here is that when people lack knowledge about the proper operation of some aspect of a machine, they are apt to derive the operation by analogy with other, similar aspects of the device. The "derivation" may be unconscious, and it can influence behavior without the person realizing that it is happening. Forming conclusions from the relationships of one system to another is a common and powerful method of human thought, but it can lead to error if the mapping from one domain onto the other is not consistent (Lakoff & Johnson, 1981; Gentner, 1980).

There are cases where lack of consistency seems desirable, and it is put into the design deliberately and with careful thought. This usually occurs when the normal sequence for an operation is long and tedious, and when such an operation is to be performed frequently it seems desirable to provide shortcuts. Similarly, the default state of an instrument or control is sometimes made inconsistent with that of other instruments or controls because experience shows that the different defaults simplify some forms of operations. Nonetheless, these inconsistencies lead to errors (and to difficulty in learning). One solution is to make command structure (and instrument format) consistent, even at the cost of some inefficiency is usage. A better solution would be to re-design the entire system so as to yield both consistency and ease of operation.

<u>Capture Errors Imply the Need for Better Feedback</u>

A capture error occurs when there is overlap in the sequence required for the performance of two different actions, especially when one is done considerably more frequently than the other. In the course of attempting the infrequent one, the more common act gets done instead. A capture error with the "vi" text editor on the Berkeley Release of the UNIX operating system occurs when attempting to write out a file. The command ":w" means to write the file, ":q" quits the editor (if the text has not been modified since the last writing of the file) and the combined sequence ":wq" writes, then quits. Because ":wq" is such a convenient operation, many people use it regularly as their way of finishing a day's session, and so it soon becomes an automatic command, with the status of a single operation rather than of two sequentially combined commands. However, as a result, at times when one wishes simply to write the file and continue with the editing, one finds oneself out of the editor and back in the operating system: by a capture error, the sequence ":wq" was typed instead of the simpler sequence ":w".

One possible way of avoiding this class of error is to minimize overlapping sequences, but this may not be possible, especially when the infrequent action sequence is simply a modification of the frequent one. In the case of "vi" if ":wq" were taken over by some other command (e.g., in newer versions of the system, "ZZ" is equivalent to ":wq") the capture error should disappear, as the two different commands — ":w" and "ZZ" have no parts in common.

A second way of avoiding the error is to try to catch it where it occurs. The error occurs at the critical place where the sequences deviate, so it is here that the problem must be faced. If the system knows what the intention of the user is (perhaps by requiring the user to indicate the overall intention), it could be designed so that at the critical choice point the proper path was flagged or in some other way brought to the attention of the operator. In addition, sufficient feedback about the state of the system should be provided to provide reminders as to the deviation from the intention. A major issue here is simply

to know the critical place at which the errors occur so that remedial action can be built into the system at that critical point.

Activation Issues Suggest the Importance of Displaying the Options and of Providing Feedback

Activation errors are of two classes; inappropriate actions get performed and appropriate actions fail to get done. The former occurs when an inappropriate action sequence is activated either by being related to desired sequences (as in the capture error) or through events in the world ("data-driven activation"). The failure to carry out an action usually occurs due to memory failure, and this can occur when events intercede between the time of preparing an intention and the time at which the act should be performed. Various memory aids seem essential to prevent the latter. The first form of activation error may very well not be preventable. In this case, the system should be designed to be tolerant of them.

People Will Make Errors, So Make the System Insensitive to Them

The analysis of errors provides one set of considerations for the construction of a system that might minimize errors. There are several other factors that should be considered as well. First, people will make errors, even in the best designed systems, even with the best of training and best of motivations. So, a corollary of the attempt to minimize errors is that one should try to minimize the effect of an error. This means that actions should be reversible, at least as much as is possible. Some things, of course, once performed, are irrevocable. Actions that can lead to difficulty should be difficult to do, perhaps requiring a set of steps (as in the release of "safeties" required when the pilot wishes to eject from a military airplane), or at least, requiring a confirmation, as when requesting that all files on a computer directory be destroyed.

It is not sufficient to ask the user to confirm that a particular action sequence is wanted, because if confirmation is routinely asked for (and if the usual response is "yes") the confirmation itself becomes an automatically invoked component of the command sequence. Thus, if the command is given in error, it is likely to have the confirmation invoked as part of the same error; in our experience, the confirmation is as apt to be in error as the original command. The point is that disastrous commands should be difficult to carry out, and confirmations of the validity of the command may not offer sufficient difficulty to be a satisfactory safeguard.

Sometimes the command need only act as if it were done, but does not in fact have to be done. Consider the command to delete files from the system; the system could claim to have removed the file, but in fact put them away on some temporary location so that they can be recovered if later their "deletion" was discovered to have been an error (real deletion can be done on an infrequent basis, say after a lapse of several hours or

basis, say after a lapse of several hours or
days). In Interlisp (Teitelman & Masinter, 1981) operations may be "undone," even operations such as writing on or destroying files.

Lessons

These simple observations lead us to some conclusions about system design. Obviously, this analysis only takes us part of the way toward a set of design principles. An analysis of errors can only get at some classes of problems, and there may not always be general rules applicable to the issues. These analyses must be supplemented with other methods, including an understanding of the nature of a person's mental image of the system that is being used and an understanding of the human information processing capability of the user. Meanwhile, the analyses presented here do make several points that are useful to summarize:

* Feedback: The state of the system should be clearly available to the user, ideally in a form that is unambiguous and that makes the set of options readily available so as to avoid mode errors.

* Similarity of response sequences: Different classes of actions should have quite dissimilar command sequences (or menu patterns) so as to avoid capture and description errors.

* Actions should be reversible (as much as possible) and where both irreversible and of relatively high consequence, they should be difficult to do, thereby preventing unintentional performance.

* Consistency of the system: The system should be consistent in its structure and design of command so as to minimize memory problems in retrieving the operations.

These considerations, coupled with similar analyses of the properties of the users' mental models of the system lead to other sets of rules for performance, including the notion of a <u>system image</u>, which should be the first thing set up by the designer, and with all commands, feedback, and instruction designed to be consistent with that system image. However, this is another story, one that too is but in the early stages of development, and which is not fully ready to be discussed here.

References

Card, S., Moran, T., & Newell, A., Applied information-processing psychology: The human-computer interface. Hillsdale, N.J.: Erlbaum Associates, 1982.

Gentner, D. The Structure of Analogical Models in Science: Technical Report 4451. Cambridge, Mass: Bolt, Beranek and Newman, 1980.

Gentner, D. & Stevens, A. (Eds.), Mental models. Hillsdale, N.J.: Erlbaum Associates, 1982 (in press).

Lakoff, G. & Johnson, M. Metaphors we live by. Chicago: University of Chicago Press, 1980.

Lockheed Missiles & Space Company. Human factors review of nuclear power plant control room design. EPRI NP-309-SY. Palo Alto, California.: Electric Power Research Institute, 1976.

National Transportation Safety Board. Aircraft incident report: Aero Mexico DC-10-30, XA-DUH over Luxembourg, Europe, November 11, 1979. (NTSB-AAR-80-10). Washington, D.C.: National transportation safety board, November, 1980. (a) (Report available from National Technical Information Service.)

National Transportation Safety Board. Special investigation report: Design-induced landing gear retraction accidents in Beech Baron, Bonanza, and other light aircraft. (NTSB-SR-80-1). Washington, D.C.: National Transportation Safety Board, June, 1980. (b) (Report available from National Technical Information Service.)

Norman, D. A. Categorization of action slips. Psychological Review, 1981, 88, 1-15.

Norman, D. A., Some observations on mental models. In D. Gentner and A. Stevens (Eds.), Mental models. Hillsdale, N.J.: Erlbaum Associates, 1982 (in press).

Teitelman, W., & Masinter, L., The interlisp programming environment. Computer, 1981, 14, (April, No. 4), 25 - 33.

The system design is elegant but the user interface is not.

DATAMATION
NOVEMBER 1981
VOLUME 27 NUMBER 12

THE TROUBLE WITH UNIX

by Donald A. Norman

UNIX is a highly touted operating system. Developed at the Bell Telephone Laboratories and distributed by Western Electric, it has become a standard operating system in universities, and it promises to become a standard for micro and mini systems in homes, small businesses, and schools. But for all of its virtues as a system—and it is indeed an elegant system—UNIX is a disaster for the casual user. It fails both on the scientific principles of human engineering and even in just *plain common sense*.

If UNIX is really to become a general system, then it has got to be fixed. I urge correction to make the elegance of the system design be reflected as friendliness towards the user, especially the casual user. Although I have learned to get along with the vagaries of UNIX's user interface, our secretarial staff persists only because we insist.

And even I, a heavy user of computer systems for 20 years, have had difficulties: copying the old file over the new, transferring a file into itself until the system collapsed, and removing all the files from a directory simply because an extra space was typed in the argument string. The problem is that UNIX fails several simple tests.

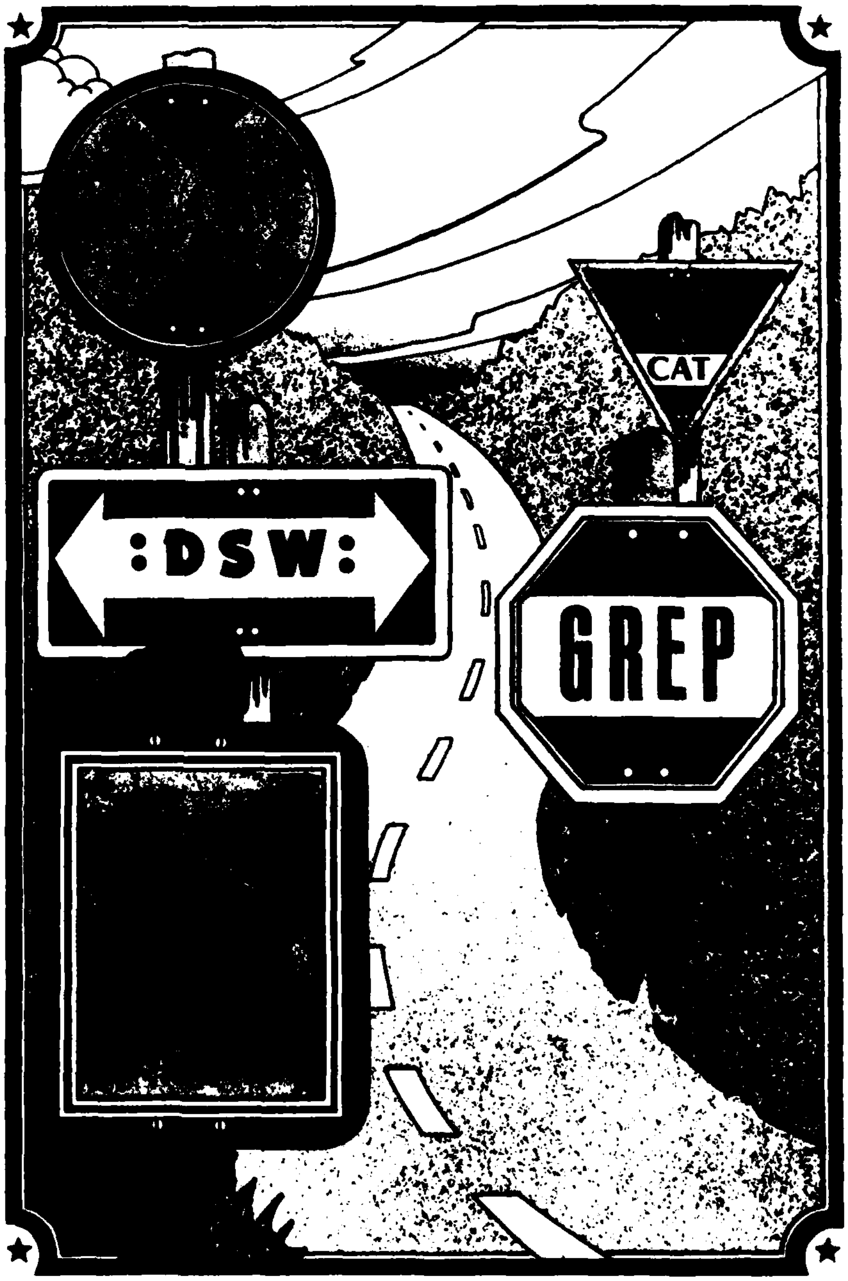
Consistency: Command names, language, functions, and syntax are inconsistent.

Functionality: The command names, formats, and syntax seem to have no relationship to their functions.

Friendliness: UNIX is a recluse, hidden from the user, silent in operation. The lack of interaction makes it hard to tell what state the system is in, and the absence of mnemonic structures puts a burden on the user's memory.

What is good about UNIX? The system design, the generality of programs, the file structure, the job structure, the powerful operating system command language (the "shell"). Too bad the concern for system design was not matched by an equal concern for the human interface.

One of the first things you learn when you start to decipher UNIX is how to list the contents of a file onto your terminal. Now this sounds straight forward enough, but in UNIX



WHAT IS UNIX?

UNIX is an operating system developed by Dennis Ritchie and Ken Thompson of Bell Laboratories. UNIX is trademarked by Bell Labs and is available under license from Western Electric. Although UNIX is a relatively small operating system, it is quite powerful and general. It has found considerable favor among programming groups, especially in universities, where it is primarily used with DEC computers—various versions of the DEC PDP-11 and the VAX. The operating system and its software are written in a high level programming language called C, and most of the source code and documentation is available on-line. For programmers, UNIX is easy to understand and to modify.

For the nonexpert programmer, the important aspect of UNIX is that it is constructed out of a small, basic set of concepts and programming modules, with a flexible method for interconnecting existing modules to make new functions. All system objects—including all I/O channels—look like files. Thus, it is possible to cause input and output for almost any program to be taken from or to go to files, terminals, or other devices, at any time, without any particular planning on the part of the module writer. UNIX has a hierarchical file structure. Users can add and delete file directories at will and then "position" themselves at different locations in the resulting hierarchy to make it easy to manipulate the files in the neighborhood.

The command interpreter of the operating system interface (called the "shell") can take its input from a file, which means that it is possible to put frequently used sequences of commands into a file and then invoke that file (just by typing its name), thereby executing the command strings. In this way, the user can extend the range of commands that are readily available. Many users end up with a large set of specialized shell command files. Because the shell includes facilities for passing arguments, for iterations, and for conditional operations, these "shell programs" can do quite a lot, essentially calling upon all system resources (including the editors) as subroutines. Many nonprogrammers have discovered that they can write powerful shell programs, thus significantly enhancing the power of the overall system.

By means of a communication channel known as a pipe, the output from one program can easily be directed (piped) to the input of another, allowing a sequence of programming modules to be strung together to do some task that in other systems would have to be done by a special purpose program. UNIX does not provide special purpose programs. Instead, it attempts to provide a set of basic software tools that can be strung together in flexible ways using I/O redirection, pipes, and shell programs. Technically, UNIX is just the operating system. However, because of the way the system has been packaged, many people use the name to include all of the programs that come on the distribution tape. Many people have found it easy to modify the UNIX system and have done so, which has resulted in hordes of variations on various kinds of computers. The "standard UNIX" discussed in the article is BTL UNIX Version 6 (May 1975). The Fourth Berkeley Edition of UNIX is more or less derived from BTL UNIX Version 7 (September 1978), with considerable parallel development at the University of California, Berkeley and some input from other BTL UNIX versions. I am told that some of the complaints in the article have been fixed; however, Version 6 is still used by many people.

The accompanying article is written with heavy hand, and it may be difficult to discern that I am a friend of UNIX. The negative tone should not obscure the beauty and power of the operating system, file structure, and the shell. UNIX is indeed a superior operating system. I would not use any other. Some of the difficulties detailed result from the fact that many of the system modules were written by the early users of UNIX, not by the system designers; a lot of individual idiosyncrasies have gotten into the system. It is my hope that the positive aspects of the article will not be overlooked. They can be used by all system designers, not just by those working on UNIX. Some other systems need these comments a lot more than does UNIX.

—D.A.N.

even this simple operation has its drawbacks. Suppose I have a file called "testfile." I want to see what is inside of it. How would you design a system to do it? I would have written a program that listed the contents onto the terminal, perhaps stopping every 24 lines if you had signified that you were on a display terminal with only a 24-line display. UNIX, however, has no basic listing command, and instead uses a program meant to do something else.

Thus if you want to list the contents of a file called "HappyDays," you use the command named "cat":

cat HappyDays

Why cat? Why not? After all, as Humpty Dumpty said to Alice, who is to be the boss, words or us? "Cat," short for "concatenate" as in, take file1 and concatenate it with file2 (yielding one file, with the first part file1, the second file2) and put the result on the "standard output" (which is usually the terminal):

cat file1 file2

Obvious, right? And if you have only one file, why cat will put it on the standard output—the terminal—and that accomplishes the goal (except for those of us with video terminals, who watch helplessly as the text goes streaming off the display)

The UNIX designers believe in the principle that special-purpose functions can be avoided by clever use of a small set of system primitives. Why make a special function when the side effects of other functions will do what you want? Well, for several reasons:

- Meaningful terms are considerably easier to learn than nonmeaningful ones. In computer systems, this means that names should reflect function, else the names for the function will be difficult to recall.
- Making use of the side effects of system primitives can be risky. If cat is used unwisely, it will destroy files (more on this in a moment).
- Special functions can do nice things for users, such as stop at the end of screens, or put on page headings, or transform nonprinting characters into printing ones, or get rid of underlines for terminals that can't do that.

Cat, of course, won't stop at terminal or page boundaries, because doing so would disrupt the concatenation feature. But still, isn't it elegant to use cat for listing? Who needs a print or a list command? You mean "cat" isn't how you would abbreviate concatenate? It seems so obvious, just like:

FUNCTION	UNIX COMMAND NAME
c compiler	cc
change working directory	chdir
change password	passwd
concatenate	cat
copy	cp
date	date
echo	echo
editor	ed
link	ln
move	mv
remove	rm
search file for pattern	grep

Notice the lack of consistency in forming the command name from the function. Some names are formed by using the first two consonants of the function name. Editor, however, is "ed," concatenate is "cat," and "date" and "echo" are not abbreviated at all. Note how useful those two-letter abbreviations are. They save almost 400 milliseconds per command.

Similar problems exist with the names of the file directories. UNIX is a file-oriented system, with hierarchical directory structures, so the directory names are very important. Thus, this paper is being written on a file named "unix" and whose "path" is csl norman papers CogEngineering unix. The name of the top directory is " ", and csl, norman, papers, and CogEngineering are the names of directories hierarchically placed beneath " ". Note that the symbol " " has two meanings the name of the top level directory and the symbol that separates levels of the directories. This is very difficult to justify to new users. And those names. he directory for "users" and "mount" are called, of course,

After all, as Humpty Dumpty said to Alice, who is to be the boss, words or us?

"usr" and "mnt." And there are "bin," "lib," and "tmp" (binary, library, and temp). UNIX loves abbreviations, even when the original name is already very short. To write "user" as "usr" or "temp" as "tmp" saves an entire letter: a letter a day must keep the service person away. But UNIX is inconsistent: it keeps "grep" at its full four letters, when it could have been abbreviated as "gr" or "gp." (What does grep mean? "Global REgular expression, Print"—at least that's the best we can invent; the manual doesn't even try. The name wouldn't matter if grep were something obscure, hardly ever used, but in fact it is one of the more powerful, frequently used string processing commands.)

LIKE CAT? THEN TRY DSW

Another important routine goes by the name of "dsw." Suppose you accidentally create a file whose name has a nonprinting character in it. How can you remove it? The command that lists the files on your directory won't show nonprinting characters. And if the character is a space (or worse, a "*"), "rm" (the program that removes files) won't accept it. The name "dsw" was evidently written by someone at Bell Labs who felt frustrated by this problem and hacked up a quick solution. Dsw goes to each file in your directory and asks you to respond "yes" or "no," whether to delete the file or keep it.

How do you remember dsw? What on earth does the name stand for? The UNIX people won't tell; the manual smiles the wry smile of the professional programmer and says, "The name dsw is a carryover from the ancient past. Its etymology is amusing." Which operation takes place if you say "yes"? Why, the file is deleted of course. So if you go through your files and see important-file, you nod to yourself and say, yes, I had better keep that one. You type in "yes," and destroy it forever. There's no warning; dsw doesn't even document itself when it starts, to remind you of which way is which. Berkeley UNIX has finally killed dsw, saying "This little known, but indispensable facility has been taken over . . ." That is a fitting commentary on standard UNIX: a system that allows an "indispensable facility" to be "little known."

The symbol "*" means "glob" (a typical UNIX name: the name tells you just what it does, right?). Let me illustrate with our friend, "cat." Suppose I want to collect a set of files named paper.1 paper.2 paper.3 and paper.4 into one file. I can do this with cat:

cat paper.1 paper.2 paper.3 paper.4> newfilename

UNIX provides "glob" to make the job even easier. Glob means to expand the filename by examining all files in the directory to find all that fit. Thus, I can redo my command as

cat paper*>newfilename

where paper* expands to {paper.1 paper.2 paper.3 paper.4}. This is one of the typical virtues of UNIX; there are a number of quite helpful functions. But suppose I had decided to name this new file "paper.all"—pretty logical name.

cat paper*>paper.all

Disaster. In this case, paper* expands to paper.1 paper.2 paper.3 paper.4 paper.all, and so I am filling up a file from itself:

cat paper.1 paper.2 paper.3 paper.4 paper.all>paper.all

Eventually the file will burst. Does UNIX check against this, or at least give a warning? No such luck. The manual doesn't alert users to this either, although it does warn of another, related infelicity: "Beware of 'cat a b > a' and 'cat b a > a', which destroy the input files before reading them." Nice of them to tell us.

The command to remove all files that start with the word "paper"

rm paper*

becomes a disaster if a space gets inserted by accident:

rm paper *

for now the file "paper" is removed, as well as every file in the entire directory (the power of glob). Why is there not a check against such things? I finally had to alter my version of rm so that when I said to remove files, they were moved to a special directory named "deleted" and preserved there until I logged off, leaving me lots of time for second thoughts and catching errors. This illustrates the power of UNIX: what other operating system would make it so easy for someone to completely change the operation of a system command? It also illustrates the trouble with UNIX: what other operating system would make it so necessary to do so? (This is no longer necessary now that we use Berkeley UNIX—more on this in a moment.)

THE SHY TEXT EDITOR

The standard text editor is called Ed. I spent a year using it as an experimental vehicle to see how people deal with such confusing things. Ed's major property is his shyness; he doesn't like to talk. You invoke Ed by saying, reasonably enough, "ed." The result is silence: no response, no prompt, no message, just silence. Novices are never sure what that silence means. Ed would be a bit more likable if he answered, "thank you, here I am," or at least produced a prompt character, but in UNIX silence is golden. No response means that everything is okay; if something had gone wrong, it would have told you.

Then there is the famous append mode error. To add text into the buffer, you have to enter "append mode." To do this, you simply type "a," followed by RETURN. Now everything that is typed on the terminal goes into the buffer. (Ed, true to form, does not inform you that it is now in append mode: when you type "a" followed by "RETURN" the result is silence.) When you are finished adding text, you are supposed to type a line that "contains only a . on it." This gets you out of append mode.

Want to bet on how many extra periods got inserted into text files, or how many commands got inserted into texts, because the users thought that they were in command mode and forgot that they had not left append mode? Does Ed tell you when you have left append mode? Hah! This problem is so obvious that even the designers recognized it, but their reaction, in the tutorial introduction to Ed, was merely to note wryly that even experienced programmers make this mistake. While they may be able to see humor in the problem, it is devastating to the beginning secretary, research assistant or student trying to use UNIX as a word processor, an experimental tool, or just to learn about computers.

How good is your sense of humor? Suppose you have been working on a file for an hour and then decide to quit work, exiting Ed by saying "q." The problem is that Ed would promptly quit. Woof, there went your last hour's work. Gone forever. Why, if you had wanted to save it you would have said so, right? Thank goodness for all those other people across the country who immediately rewrote the text editor so that we normal people (who make errors) have some other choices besides Ed, editors that tell you politely when they are working, that tell you if they are in append or command mode, and that don't let you quit without saving your file unless you are first warned, and then only if you say you really mean it.

As I wrote this paper I sent out a message on our networked message system and asked my colleagues to tell me of their favorite peeves. I got a lot of responses, but there is no need to go into detail about them; they all have much the same flavor, mostly commenting about the lack of consistency and the lack of interactive feedback. Thus, there is no standardization of means to exit programs (and because the "shell" is just another program as far as the system is concerned, it is very easy to log yourself off the system by accident). There are very useful pattern matching features (such as the "glob" function), but the shell and the different programs use the symbols in inconsistent ways. The UNIX copy command (cp) and the related C programming language "stringcopy" (strcpy) reverse the meaning of their arguments, and UNIX move (mv) and copy (cp) operations will destroy existing files without any warning. Many programs take special "argument flags" but the manner of specifying the flags is inconsistent, varying

Ed's major property is his shyness; he doesn't like to talk.

ANOTHER VIEW

Prof. Norman praises the UNIX system design but makes a number of caustic remarks about command names and other aspects of the human interface. These might be ignored, since he has no experimental tests to justify them; or they might even be taken as flattery of UNIX, since he does not name any system he likes better; but some of his comments are worth discussing.

Most of the command names Norman points to are indeed strange; some, such as dsw, were removed several years ago (by the way, to repair the discourtesy of the manual, dsw meant "delete from switches"). However, it is not clear that it makes much difference what the command names are. T. K. Landauer, K. Galotti, and S. Hartwell recently tried teaching people a version of the editor in which "append," "delete," and "substitute" were called "allege," "cypher," and "deliberate." It didn't seem to have much effect on learning time, and afterwards the users would say things like "I alleged three lines and deliberated a comma on the last one" just like subjects who had learned the ordinary version of the editor ("A Computer Command By Any Other Name: *A Study of Text Editing Terms*," available from the authors at Bell Labs.)

In addition to the amusing but secondary discussion of command names, Prof. Norman does raise some significant issues: (1) whether systems should be verbose or terse; (2) whether they should have a few general commands or many special-purpose ones; and (3) whether they should try to anticipate typical mistakes. Experimental results on these issues would be welcome; meanwhile, the armchair evidence is not all on one side.

UNIX is undoubtedly near an extreme of terseness, partly because it was originally designed for slow hardcopy terminals. However, the terseness is very valuable when connecting processes. If the command that lists the logged-on users prints a heading above the list, you can't tell how many users are on by feeding the command output to a line counter. If the editor types acknowledgments now and then, its output may not be directly usable as input somewhere else. Of course, you could feed it through something which strips off the extra remarks, but presumably that program would add its own chatty messages.

Prof. Norman complains about using "cat" for a command which prints files, rather than having a special-purpose command for the purpose (there is one, by the way: "pg"). Having a few general-purpose commands is a definite aid to system learning. In practice, it is not the novices who use the alternatives to "cat"; it is the experts, who want something better adapted to their special needs and are willing to learn another command. In general, people are quite good at recognizing special uses of commands in context, probably because it is a lot like things they have to do every day in English. To take an analogy from programming languages, one doubts that Prof. Norman would advocate a separate operator for "+" in integer arithmetic and "+" in floating point arithmetic. There are many advantages to a small, general-purpose set of commands. Having only one way to do any given task minimizes software maintenance while maximizing the ability of two users to help each other with advice. But this implies that whenever a general command and a specific command do the same thing, the specific command should be removed. It would be a definite service if the "cognitive engineers" could tell us how many commands are reasonable, to give some guidance on, for example, whether "merge" should be a separate command or an option on "sort" (on UNIX it is a sort option) and whether the terminal drivers should be separate commands or options on a graphics output command (on UNIX they are separate). The best rule of thumb we have today is that designing the system so that the manual will be as short as possible minimizes learning effort.

Prof. Norman seems to think that the computer should try to anticipate user problems, and refuse commands that appear dangerous. The computer world is undoubtedly moving in this direction; strong typing in programming languages is a good example. The "ed" editor has warned for some years if the user tries to quit without writing a file. The "vi" editor has an "undo" feature, regardless of the complexity of the command which has been executed. Such a facility is undoubtedly the best solution. It lets the user recognize his mistakes and back out of them, rather than expecting the system to foresee them. It is really not possible to anticipate the infinite variety of possible user mistakes; as every programmer who has ever debugged anything knows, it is hard enough to deal with the correct inputs to a program. Human hindsight is undoubtedly better than machine foresight.

A large number of Prof. Norman's comments are pleas for consistency. UNIX has grown more than it has been built, with many people from many places tossing software into the system. The ability of the system to accept commands so easily is one of its main strengths. However, it results in command names like "finger" for what Bell Labs called "whois" (identify a user) and "more," "cat," or "pg" for what Prof. Norman would rather call "list." The thought of a UNIX Command Standardization Committee trying to impose rules on names is a frightening alternative. Much of the attractiveness of UNIX derives from its hospitality to new commands and features. This has also meant a diversity of names and styles. To some of us this diversity is attractive, while to others the diversity is frustrating, but to hope for the hospitality without the diversity is unrealistic.

—Michael Lesk
Bell Labs
Murray Hill, N.J.

from program to program.

The version of UNIX I now use is called the Fourth Berkeley Edition for the VAX, distributed by Joy, Babaoglu, Fabry, and Sklower at the University of California, Berkeley (henceforth, Berkeley UNIX). This is both good and bad.

Among the advantages: History lists, aliases, a richer and more intelligent set of system programs (including a list program, an intelligent screen editor, an intelligent set of routines for interacting with terminals according to their capabilities), and a job control that allows one to stop jobs right in the middle, start up new ones, move things from background to foreground (and vice versa), examine files, and then resume jobs. The shell has been amplified to be a more powerful programming language, complete with file handling capabilities, if—then—else statements, while, case, and other goodies of structured programming (see box, p. 00).

Aliases are worthy of special comment. Aliases let users tailor the system to their own needs, naming things in ways they can remember; names you devise yourself are easier to recall than names provided to you. And aliases allow abbreviations that are meaningful to the individual, without burdening everyone else with your cleverness or difficulties.

To work on this paper, I need only type the word "unix," for I have set up an alias called "unix" that is defined to be equal to the correct command to change directories, combined with a call to the editor (called "vi" for "visual" on this system) on the file:

alias unix "chdir /cs1/norman/papers/CogEngineering; vi unix"

These Berkeley UNIX features have proven to be indispensable: the people in my laboratory would probably refuse to go back to standard UNIX.

The bad news is that Berkeley UNIX is jury-rigged on top of regular UNIX, so it can

There are lots of aids to memory that can be provided, but the most powerful of all is understanding.

only patch up the faults: it can't remedy them. Grep is not only still grep, but there is an egrep and an fgrep.

And the generators of Berkeley UNIX have their problems: if Bell Labs people are smug and lean, Berkeley people are cute and overweight. Programs are wordy. Special features proliferate. The system is now so large that it no longer fits on the smaller machines: our laboratory machine, a DEC 11/45, cannot hold the latest release of Berkeley UNIX (even with a full complement of memory and a reasonable amount of disk). I wrote this paper on a VAX.

LEARNING IS NOT EASY

Learning the system for setting up aliases is not easy for beginners, who may be the people who need them most. You have to set them up in a file called .cshrc, not a name that inspires confidence. The "period" in the filename means that it is invisible—the normal method of directory listing programs won't show it. The directory listing program, ls, comes with 19 possible argument flags, which can be used singly or in combinations. The number of special files that must be set up to use all the facilities is horrendous, and they get more complex with each new release from Berkeley.

It is very difficult for new users. The program names are cute rather than systematic. Cuteness is probably better than standard UNIX's lack of meaning, but there are limits. The listing program is called "more" (as in, "give me more"), the program that tells you who is on the system is called "finger," and a keyword help file—most helpful, by the way—is called "apropos." I used the alias feature to rename it "help."

One reader of a draft of this paper—a systems programmer—complained bitterly: "Such whining, hand-wringing, and general bitchiness will cause most people to dismiss it as over-emotional nonsense. . . . The UNIX system was originally designed by systems programmers for their own use and with no intention for others using it. Other hackers liked it so much that eventually a lot of them started using it. Word spread about this wonderful system, and the rest you probably know. I think that Ken Thompson and Dennis Ritchie could easily shrug their shoulders and say 'But we never intended it for other than our personal use.' "

This complaint was unique, and I sympathize with its spirit. It should be remembered, though, that UNIX is nationally distributed under strict licensing agreements. Western Electric's motives are not altogether altruistic. If UNIX had remained a simple experiment on the development of operating systems, then complaints could be made in a more friendly, constructive manner. But UNIX is more than that. It is taken as the very model of a proper operating system. And that is exactly what it is not.

In the development of the system aspects of UNIX, the designers have done a magnificent job. They have been creative, and systematic. A common theme runs through the development of programs, and by means of their file structure, the development of "pipes" and "redirection" of both input and output, plus the power of the iterative "shell" system-level commands, one can easily combine system level programs into self-tailored systems of remarkable power. For system programmers, UNIX is a delight. It is well structured, with a consistent, powerful philosophy of control and structure.

Why was the same effort not put into the design at the level of the user? The answer is complex, but one reason is the fact that there really are no well known principles of design at the level of the user interface. So, to remedy the harm I may have caused with my heavy-handed sarcasm, let me attempt to provide some positive suggestions based upon research conducted by myself and others into the principles of the human information processing system.

Cognitive engineering is a new discipline, so new that it doesn't exist, but it ought to. Quite a bit is known about the human information processing system, enough that we can specify some basic principles for designers. People are complex entities and can adapt to almost anything. As a result, designers often design for themselves, without regard for other kinds of users.

The three most important concepts for system design are these:

1. Be consistent. A fundamental set of principles ought to be evolved and followed consistently throughout all phases of the design.

2. Provide the user with an explicit model. Users develop mental models of the devices with which they interact. If you do not provide them with one, they will make one up themselves, and the one they create is apt to be wrong.

Do not count on the user fully understanding the mechanics of the device. Both secretaries and scientists may be ignorant of the difference between the buffer, the working memory, the working files, and the permanent files of a text editor. They are apt to believe that once they have typed something into the system, it is permanently in their files. They are apt to expect more intelligence from the system than the designer knows is there. And they are apt to read into comments (or the lack of comments) more than you have intended.

Feedback is of critical importance in helping establish the appropriate mental model and in letting the user keep its current state in synchrony with the actual system.

3. Provide mnemonic aids. For most purposes it is convenient to think of human memory as consisting of two parts: a short-term memory and a long-term memory (modern cognitive psychology is developing more sophisticated notions, but this is still a valid approximation). Five to seven items is about the limit for short-term memory. Thus, do not expect a user to remember the contents of a message for much longer than it is visible on the terminal. Long-term memory is robust, but it faces two difficulties: getting stuff in so that it is properly organized, and getting stuff out when it is needed. Learning is difficult, unless there is a good structure and it is visible to the learner.

There are lots of sensible memory aids that can be provided, but the most powerful and sensible of all is understanding. Make the command names describe the function that is desired. If abbreviations must be used, adopt a consistent policy of forming them. Do not deviate from the policy, even when it appears that a particular command warrants doing so.

System designers take note. Design the system for the person, not for the computer, *not even for yourself*. People are also information processing systems, with varying degrees of knowledge and experience. Friendly systems treat users as normal, intelligent adults who are sometimes forgetful and are rarely as knowledgeable about the world as they would like to be. There is no need to talk down to the user, nor to explain everything. But give the users a share in understanding by presenting a consistent view of the system. Their response will be your reward. *

Partial research support was provided by Contract N00014-79-C-0323, NR 157-437 with the Personnel and Training Research Programs of the Office of Naval Research, and was sponsored by the Office of Naval Research and the Air Force Office of Scientific Research. I thank the members of the LNR research group for their helpful suggestions and descriptions of misery. In particular, I wish to thank Phil Cohen, Tom Erickson, Jonathan Grudin, Henry Halff, Gary Perlman, and Mark Wallen for their analysis of UNIX. Gary Perlman and Mark Wallen provided a number of useful suggestions.

Donald A. Norman is professor of psychology and director of the program in cognitive science at the University of California, San Diego. He has degrees in electrical engineering from MIT and the University of Pennsylvania, and a doctorate in psychology from the University of Pennsylvania. He is the author of seven books, including *Human Information Processing*, Academic Press, N.Y., 1977.

"The Trouble with Networks" by Donald A. Norman. Published in Datamation, January 1982, pp. 188-192.

THE TROUBLE WITH NETWORKS

Computer networks are in everyone's future, the prophets tell us, and will bring about new styles of communication. How will the changes affect us? The following, a personal case study in the sociology of computer networks, may provide a few clues.

One weekend, when I should have been doing something else, and when I had once more made a minor error while working at my computer terminal, I collected my frustrations together into the paper "The Trouble with UNIX" (November, p. 139). Little did I know that I was thereby setting into motion a chain of events that would occupy me for many months. A draft of my article was circulated on a national computer network, and soon brought me fame and insult.

My computer is part of a hardwired campus network of computers that use the UNIX operating system. This local campus network is, in turn, part of a statewide telephone UNIX network that interconnects many campuses of the University of California. The campus network is connected to a nationwide dial-up network of UNIX users, which was started, I believe, by people at various Bell Laboratories in New Jersey and elsewhere. The campus network is also connected to the Defense Department's packet-switching network (ARPANET). In addition to distributing messages, manuscripts, programs, and documentation, the net also provides a number of "bulletin boards." These are collections of messages constructed so that people who are interested in related topics can hold discussions, ask specialized questions of one another, and pass on information thought to be of general interest.

One day, someone logged onto my computer, found the file in which the paper was located, and distributed it through the network bulletin board called "UNIX-wizards." From there it was distributed over at least one other bulletin board, and probably over many individual messages and to many different locations. Thus, the paper was sent all over the country, to large numbers of sites and possibly thousands of readers. This was done without my knowledge (and therefore, without my permission, although I would have granted permission had I been asked). In addition, the culprit managed to disguise the transmission so that the source could not be identified.

How did that happen? Well, we run an open computer facility, and users are allowed to look over other people's files. We can and do protect confidential material. The rule is that people may look freely, and if users wish some things to be private, they must protect the access to those files. The location of my paper was well known because I had asked people to read it and give me feedback. The thief was probably someone authorized to use our machine, and probably an expert systems person who had the knowledge and authorization to disguise the source of the transmission.

The first news I had that my paper had been distributed was when a local systems programmer sent me a copy of a message he was sending in response to someone at Bell Labs, agreeing with the Bell Labs person that my article was appalling. The only part of the original message from Bell Labs that I was allowed to see said, "Who is Don Norman and why is he saying those terrible things about me?"

From that point on, things got hectic. A flurry of comments appeared on several of the bulletin boards. The article was a disaster, said one. Another person agreed with many of the observations, but thought "the paper should not be a criticism of UNIX itself, but rather a criticism of how people use UNIX." Some people said the piece was correct. Others claimed it was all wrong, and that anyone who had problems with UNIX didn't deserve to be using it in the first place. Eventually, people discovered my proper network address and began sending mail directly to me, bypassing the bulletin boards. I was flooded with comments. "DATAMATION readers are typically IBM users," said one note, "and they will now say 'You see? UNIX is poorly designed, this psychologist says so. Wake me up when you have an operating system better than IBM's.' . . . I hope that both of you live happily ever after." From Texas, Utah, Toronto, from the various Bell Labs, from California, from Massachusetts—the notes kept coming. Soon I was spending over an hour each morning just reading the previous day's accumulation and answering them. When I printed out all the messages that I had received on a hardcopy terminal, it took 32 single-spaced pages.

The majority of these comments were laudatory. Several people wanted advice on systems they were working on. I had useful interchanges with them and, in the process, clarified my own understanding of the issues. A manager of a major system—call it System X—gave me a computer account on X (and promised to send the manuals), asking me to do a similar analysis on it so that his team could improve it. He did lay down the restriction that I not write a new article called "The Truth About System X." Although we are not yet finished with our analysis (the manuals haven't arrived yet), interaction like this is quite gratifying—the kind of thing one hopes for.

The most positive interactions, however, took place with people at Bell Laboratories—people who had been on the receiving end of my criticisms. After a somewhat hesitant start, our dialog became quite useful. We have discussed a number of issues and agreed upon some points that need further treatment. I sent them the box describing UNIX that was published alongside my article, and they rewrote it to clarify points and correct errors. They sent me the rebuttal they were writing, and I thought it a good one (it too was published alongside my article). I sent them several papers that I was working on and they sent me reprints of their published papers on UNIX. One of our current graduate students sent descriptions of the menu-driven command interpreter for the UNIX shell that he was developing, and so on. These interchanges have helped clarify issues on all sides, eventually leading all of us to a better understanding of the constraints on system design and release, and to an awareness of the needs and limitations of a wide variety of users.

(CONTINUED NEXT PAGE)

The weaknesses and the strengths of computer networking derive from the same feature: it is easy to send messages to anyone who has access to the network. Because a new message is so easily generated, it can be composed immediately upon the receipt of one that has aroused the emotions. But if a message is composed in the heat of passion, the passion may distort it, so that the result is not always as effective as one might hope.

The ease with which people can generate additions to the bulletin board and messages to others has another major drawback: electronic junk mail. Many of my colleagues and I have stopped reading the network news and bulletin boards because we cannot afford the time to do so every day. (I did not know my paper had been distributed and would not have discovered the flurry it created had others not alerted me to it.) This will be less of a problem when we have intelligent programs that can aid in browsing through tables of contents, perhaps with intelligent keyword or content specified searches to winnow through the accumulation. Perhaps we can put together some quasi-intelligent text-understanding systems that can help sort through the material. However, until something is done to improve the organization, the very success of these message systems and bulletin boards will threaten their usefulness.

Another interesting social phenomenon that may occur within an organization possessing an effective computer mail system is that people will tend to use it in preference to talking. Computer mail is much more efficient than telephone calls or visits because you can generate it whenever you wish without concern for whether the recipient is in. Similarly, the recipient can read and answer messages at leisure. It is better than postal mail or interoffice memos because it is easier, less formal, and can be almost instantaneous if the recipient wishes it to be. In our laboratory, this sometimes leads to strange behavior. It is not unheard of for one person to see another in the hall and to say "I am going to send you a message," and then go do so, forgetting that the information could simply have been spoken.

The positive side of these networks overcomes the negative. People can communicate their ideas to others across the country, quickly and effectively. In turn, the recipients can respond, criticizing, sharing, and improving the product. The network communications keep me informed on a variety of issues from substantive research topics to trip and conference schedules. I can count on my colleagues who do read the bulletin boards to alert me to relevant articles, just as I pass on the interesting messages that I receive. Small communities of people with shared concerns can quickly be formed to hold constructive discussion about an issue. The interchanges can be quite effective, in part because of the rapidity with which messages are generated and sent: it only takes a few hours for a comment to spread out over the community.

The unauthorized distribution of my paper has been a useful sociological experience—a true test case of what will indeed be in all our futures: interactive journals, computer bulletin boards, and readily available computer message systems.

—Donald A. Norman
La Jolla, California

Navy

1 Dr. Ed Aiken
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. Arthur Bachrach
Environmental Stress Program Center
Naval Medical Research Institute
Bethesda, MD 20014

1 CDR Thomas Berghage
Naval Health Research Center
San Diego, CA 92152

1 Chief of Naval Education and Training
Liason Office
Air Force Human Resource Laboratory
Flying Training Division
Williams AFB, AZ 85224

1 CDR Mike Curran
Office of Naval Research
800 N. Quincy St.
Code 270
Arlington, VA 22217

1 Dr. Pat Federico
Navy Personnel R&D Center
San Diego, CA 92152

1 Dr. John Ford
Navy Personnel R&D Center
San Diego, CA 92152

1 LT Steven D. Harris, MSC, USN
Code 6021
Naval Air Development Center
Warminster, Pennsylvania 18974

1 Dr. Patrick R. Harrison
Psychology Course Director
Leadership & Law Dept. (7b)
Div. of Professional Development
U.S. Naval Academy
Annapolis, MD 21402

1 Dr. Jim Hollan
Code 304
Navy Personnel R & D Center
San Diego, CA 92152

1 CDR Charles W. Hutchins
Naval Air Systems Command Hq
AIR-340F
Navy Department
Washington, DC 20361

1 Dr. Norman J. Kerr
Chief of Naval Technical Training
Naval Air Station Memphis (75)
Millington, TN 38054

1 Dr. William L. Maloy
Principal Civilian Advisor for
Education and Training
Naval Training Command, Code 00A
Pensacola, FL 32508

1 Capt. Richard L. Martin, USN
Prospective Commanding Officer
USS Carl Vinson (CVN-70)
Newport News Shipbuilding and Drydock Co.
Newport News, VA 23607

1 Dr. George Moeller
Head, Human Factors Dept.
Naval Submarine Medical Research Lab
Groton, CN 06340

1 Dr. William Montague
Navy Personnel R & D Center
San Diego, CA 92152

1 Commanding Officer
U.S. Naval Amphibious School
Coronado, CA 92155

1 Ted M. I. Yellen
Technical Information Office, Code 201
Navy Personnel R&D Center
San Diego, CA 92152

1 Library, Code P201L
Navy Personnel R&D Center
San Diego, CA 92152

1 Technical Director
Navy Personnel R&D Center
San Diego, CA 92152

6 Commanding Officer
Naval Research Laboratory
Code 2627
Washington, DC 20390

1 Psychologist
ONR Branch Office
Bldg 114, Section D
666 Summer Street
Boston, MA 02210

1 Psychologist
ONR Branch Office
536 S. Clark Street
Chicago, IL 60605

1 Office of Naval Research
Code 437
800 N. Quincy Street
Arlington, VA 22217

1 Office of Naval Research
Code 441
800 N. Quincy Street
Arlington, VA 22217

5 Personnel & Training Research Programs
(Code 458)
Office of Naval Research
Arlington, VA 22217

1 Psychologist
ONR Branch Office
1030 East Green St.
Pasadena, CA 91101

1 Office of the Chief of Naval Operations
Research Development & Studies Branch
(OP-115)
Washington, DC 20350

1 Lt. Frank C. Petho, MSC, USN (Ph.D)
Selection and Training Research Div.
Human Performance Sciences Dept.
Naval Aerospace Medical Research Lab.
Pensacola, FL 32508

1 Dr. Gary Poock
Operations Research Dept.
Code 55PK
Naval Postgraduate School
Monterey, CA 93940

1 Roger W. Remington, Ph.D
Code L52
NAMRL
Pensacola, FL 32508

1 Dr. Worth Scanland, Director
Research, Development, Test & Eval.
Naval Education and Training
Code N-5
NAS, Pensacola, FL 32508

1 Dr. Sam Schiflett, SY 721
Systems Engineering Test Directorate
U.S. Naval Air Test Center
Patuxent River, MD 20670

1 Dr. Robert G. Smith
Office of Chief of Naval Operations
OP-987H
Washington, DC 20350

1 Dr. Alfred F. Smode
Training Analysis & Evaluation Group
(TAEG)
Dept. of the Navy
Orlando, FL 32813

1 Dr. Richard Sorensen
Navy Personnel R&D Center
San Diego, CA 92152

1 Roger Weissinger-Baylon
Dept. of Admin. Sciences
Naval Postgraduate School
Monterey, CA 93940

1 Dr. Robert Wisher
Code 309
Navy Personnel R&D Center
San Diego, CA 92152

1 Mr John H. Wolfe
Code P310
U. S. Navy Personnel Research and
Development Center
San Diego, CA 92152

Army

1 Technical Director
U.S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Ave.
Alexandria, VA 22333

1 Mr. James Barber
HQS, Department of the Army
DAPE-ZBR
Washington, DC 20310

1 Dr. Beatrice J. Farr
U.S. Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 Dr. Michael Kaplan
U.S. Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 Dr. Milton S. Katz
Training Technical Area
U.S. Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 Dr. Harold F. O'Neil, Jr.
Attn: PERI-OK
Army Research Institute
5001 Eisenhower Ave.
Alexandria, VA 22333

1 LTC Michael Plummer
Chief, Leadership & Organizational
Effectiveness Division
Office of the Deputy Chief of Staff
for Personnel
Dept. of the Army
Pentagon, Washington DC 20301

1 Dr. Robert Sasmor
U. S. Army Research Institute for the
Behavioral and Social Sciences
5001 Eisenhower Avenue
Alexandria, VA 22333

Air Force

1 Air University Library
AUL/LSE 76/443
Maxwell AFB, AL 36112

1 Dr. Earl A. Alluisi
HQ, AFHRL (AFSC)
Brooks AFB, TX 78235

1 Dr. Genevieve Haddad
Program Manager
Life Sciences Directorate
AFOSR
Bolling AFB, DC 20332

1 Dr. Sylvia R. Mayer (TO1T)
HQ Electronic Systems Division
Hanscom AFB
Bedord, MA 02173

1 Dr. Frank Schufletowski
U.S. Air Force
ATC/XPTD
Randolph AFB, TX 78148

2 3700 TCHTW/TTGH Stop 32
Sheppard AFB, TX 76311

Marines

1 H. William Greenup
Education Advisor (E031)
Education Center, MCDEC
Quantico, VA 22134

1 Special Assistant for Marine
Corps Matters
Code 100M
Office of Naval Research
800 N. Quincy St.
Arlington, VA 22217

1 Dr. A.L. Slafkosky
Scientific Advisor (Code RD-1)
HQ, U.S. Marine Corps
Washington, DC 20380

CoastGuard

1 Chief, Psychological Research Branch
U. S. Coast Guard (G-P-1/2/TP42)
Washington, DC 20593

Other DoD

12 Defense Technical Information Center
Cameron Station, Bldg. 5
Alexandria, VA 22314
Attn: TC

1 Military Assistant for Training and
Personnel Technology
Office of the Under Secretary of Defense
for Research & Engineering
Room 3D129, The Pentagon
Washington, DC 20301

1 DARPA
1400 Wilson Blvd.
Arlington, VA 22209

Civil Govt

1 Dr. Paul G. Chapin
Linguistics Program
National Science Foundation
Washington, DC 20550

1 Dr. Susan Chipman
Learning and Development
National Institute of Education
1200 19th Street NW
Washington, DC 20208

1 William J. McLaurin
66610 Howie Court
Camp Springs, MD 20031

1 Dr. Arthur Melmed
National Institute of Education
1200 19th Street NW
Washington, DC 20208

1 Dr. Andrew R. Molnar
Science Education Dev.
and Research
National Science Foundation
Washington, D.C. 20550

1 Dr. Joseph Psotka
National Institute of Education
1200 19th St. NW
Washington, D.C. 20208

1 Dr. H. Wallace Sinaiko
Program Director
Manpower Research and Advisory Services
Smithsonian Institution
801 North Pitt Street
Alexandria, VA 22314

1 Dr. Frank Withrow
U.S. Office of Education
400 Maryland Ave. SW
Washington, D.C. 20202

1 Dr. Joseph L. Young, Director
Memory & Cognitive Processes
National Science Foundation
Washington, DC 20550

Non Govt

1 Dr. John R. Anderson
Dept. of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

1 Dr. John Annett
Dept. of Psychology
University of Warwick
Coventry CV4 7AL
England

1 1 Psychological Research Unit
Dept. of Defense (Army Office)
Campbell Park Offices
Canberra ACT 2600, Australia

1 Dr. Alan Baddeley
Medical Research Council
Applied Psychology Unit
15 Chaucer Rd.
Cambridge CB2 2EF
England

1 Dr. Patricia Baggett
Dept. of Psychology
University of Colorado
Boulder, CO 80309

1 Dr. Jonathan Baron
Dept. of Psychology
University of Pennsylvania
3813-15 Walnut St. T-3
Philadelphia, PA 19104

1 Mr Avron Barr
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Jackson Beatty
Department of Psychology
University of California
Los Angeles, CA 90024

1 Liaison Scientists
Office of Naval Research
Branch Office, London
Box 39 FPO New York 09510

1 Dr. Lyle Bourne
Department of Psychology
University of Colorado
Boulder, CO 80309

1 Dr. John S. Brown
XEROX Palo Alto Research Center
3333 Coyote Road
Palo Alto, CA 94304

1 Dr. Bruce Buchanan
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. C. Victor Bunderson
WICAT INC.
University Plaza Suite 10
1160 So. State St.
Orem, UT 84057

1 Dr. Pat Carpenter
Dept. of Psychology
Carnegie-Mellon University
Pittsburgh, PA 15213

1 Dr. John B. Carroll
Psychometric Lab
Univ. of No. Carolina
Davie Hall 013A
Chapel Hill, NC 27514

1 Charles Myers Library
Livingstone House
Livingstone Road
Stratford
London E15 2LJ
ENGLAND

1 Dr. William Chase
Dept. of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

1 Dr. Micheline Chi
Learning R & D Center
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

1 Dr. William Clancey
Department of Computer Science
Stanford University
Stanford, CA 94305

1 Dr. Allan M. Collins
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

1 Dr. Lynn A. Cooper
LRDC
University of Pittsburgh
3939 O'Hara St.
Pittsburgh, PA 15213

1 Dr. Meredith P. Crawford
American Psychological Association
1200 17th Street, N.W.
Washington, DC 20036

1 Dr. Kenneth B. Cross
Anacapa Sciences, Inc.
P.O. Drawer Q
Santa Barbara, CA 93102

1 Dr. Diane Damos
Arizona State University
Tempe, AZ 85281

1 Dr. Emmanuel Donchin
Department of Psychology
University of Illinois
Champaign, IL 61820

1 LCOL J. C. Eggenberger
Directorate of Personnel Applied Research
National Defence HQ
101 Colonel by Drive
Ottawa, Canada K1A 0K2

1 ERIC Facility-Acquisitions
4833 Rugby Avenue
Bethesda, MD 20014

1 Dr. A. J. Eschenbrenner
Dept. E422, Bldg. 81
McDonnell Douglas Astronautics Co.
P.O.Box 516
St. Louis, MO 63166

1 Dr. Ed Feigenbaum
Dept. of Computer Science
Stanford University
Stanford, CA 94305

1 Mr. Wallace Feurzeig
Bolt Beranek & Newman, Inc.
50 Moulton St.
Cambridge, MA 02138

1 Univ. Prof. Dr. Gerhard Fischer
Liebiggasse 5/3
A 1010 Vienna
AUSTRIA

1 Dr. John R. Frederiksen
Bolt Beranek & Newman
50 Moulton Street
Cambridge, MA 02138

1 Dr. Alinda Friedman
Dept. of Psychology
University of Alberta
Edmonton, Alberta
Canada T6G 2E9

1 Dr. R. Edward Geiselman
Dept. of Psychology
University of California
Los Angeles, CA 90024

1 Dr. Robert Glaser
LRDC
University of Pittsburgh
3939 O'Hara St.
Pittsburgh, PA 15213

1 Dr. Daniel Gopher
Industrial & Management Engineering
Technion-Israel Institute of Technology
Haifa
ISRAEL

1 Dr. James G. Greeno
LRDC
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

1 Dr. Harold Hawkins
Department of Psychology
University of Oregon
Eugene OR 97403

1 Dr. Barbara Hayes-Roth
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

1 Dr. Frederick Hayes-Roth
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406

1 Dr. James R. Hoffman
Dept. of Psychology
University of Delaware
Newark, DE 19711

1 Dr. Kristina Hooper
Clark Kerr Hall
University of California
Santa Cruz, CA 95060

1 Dr. Earl Hunt
Dept. of Psychology
University of Washington
Seattle, WA 98015

1 Dr. Steven W. Keele
Dept. of Psychology
University of Oregon
Eugene, OR 97403

1 Dr. David Kieras
Dept. of Psychology
University of Arizona
Tuscon, AZ 85721

1 Dr. Walter Kintsch
Dept. of Psychology
University of Colorado
Boulder, CO 80302

1 Dr. Stephen Kosslyn
Harvard University
Department of Psychology
33 Kirkland St.
Cambridge, MA 02138

1 Dr. Marcy Lansman
Dept. of Psychology
NI-25
University of Washington
Seattle, WA 98195

1 Dr. Jill Larkin
Dept. of Psychology
Carnegie Mellon University
Pittsburgh, PA 15213

1 Dr. Alan Lesgold
Learning R & D Center
University of Pittsburgh
Pittsburgh, PA 15260

1 Dr. Michael Levine
Dept. of Educational Psychology
210 Education Bldg.
University of Illinois
Champaign, IL 61801

1 Dr. Allen Munro
Behavioral Technology Laboratories
1845 Elena Ave., Fourth Floor
Redondo Beach, CA 90277

1 Committee on Human Factors
JH 811
2101 Constitution Ave. NW
Washington, DC 20418

1 Dr. Seymour A. Papert
Massachusetts Institute of Technology
Artificial Intelligence Lab
545 Technology Square
Cambridge, MA 02139

1 Dr. James A. Paulson
Portland State University
P.O. Box 751
Portland, OR 97207

1 Dr. James W. Pellegrino
University of California,
Santa Barbara
Dept. of Psychology
Santa Barbara, CA 93106

1 Mr. Luigi Petrullo
2431 N. Edgewood Street
Arlington, VA 22207

1 Dr. Martha Polson
Department of Psychology
Campus Box 346
University of Colorado
Boulder, CO 80309

1 Dr. Peter Polson
Dept. of Psychology
University of Colorado
Boulder, CO 80309

1 Dr. Steven E. Poltrock
Dept. of Psychology
University of Denver
Denver, CO 80208

1 Dr. Mike Posner
Department of Psychology
University of Oregon
Eugene, OR 97403

1 Dr. Diane M. Ramsey-Klee
R-K Research & System Design
3947 Ridgemont Drive
Malibu, CA 90265

1 Dr. Fred Reif
SESAME
c/o Physics Dept.
University of California
Berkeley, CA 94720

1 Dr. Lauren Resnick
LRDC
University of Pittsburgh
3939 O'Hara Street
Pittsburgh, PA 15213

1 Mary Riley
LRDC
University of Pittsburgh
3939 O'Hara St.
Pittsburgh, PA 15213

1 Dr. Andrew M. Rose
American Institutes for Research
1055 Thomas Jefferson St. NW
Washington, DC 20007

1 Dr. Ernst Z. Rothkopf
Bell Laboratories
600 Mountain Ave.
Murray Hill, NJ 07974

1 Dr. David Rumelhart
Center for Human Information
Processing
University of Calif. San Diego
La Jolla, CA 92093

1 Dr. Walter Schneider
Dept. of Psychology
University of Illinois
Champaign, IL 61820

1 Dr. Robert J. Seidel
Instructional Technology Group
HUMRRO
300 N. Washington St.
Alexandria, VA 22314

1 Committee on Cognitive Research
c/o Dr. Lonnie R. Sherrod
Social Science Research Council
605 Third Ave.
New York, NY 10016

1 Dr. David Shucard
Brain Sciences Labs
National Jewish Hospital
Research Center
National Asthma Center
Denver, CO 80206

1 Dr. Edward Smith
Bolt, Beranek & Newman, Inc.
50 Moulton St.
Cambridge, MA 02138

1 Dr. Richard Snow
School of Education
Stanford University
Stanford, CA 94305

1 Dr. Robert Sternberg
Dept. of Psychology
Yale University
Box 11A, Yale Station
New Haven, CT 06520

1 Dr. Albert Stevens
Bolt Beranek & Newman, Inc.
50 Moulton Street
Cambridge, MA 02138

1 David E. Stone, Ph.D.
Hazeltine Corporation
7680 Old Springhouse Rd.
McLean, VA 22102

1 Dr. Patrick Suppes
Institute for Mathematical Studies in
the Social Sciences
Stanford University
Stanford, CA 94305

1 Dr. Kikumi Tatsuoka
Computer Based Education Research
Laboratory
252 Engineering Research Laboratory
University of Illinois
Urbana, IL 61801

1 Dr. John Thomas
IBM Thomas J. Watson Research Center
P.O. Box 218
Yorktown Heights, NY 10598

1 Dr. Perry Thorndyke
The Rand Corp.
1700 Main St.
Santa Monica, CA 90406

1 Dr. Douglas Towne
University of So. Calif.
Behavioral Technology Labs
1845 S. Elena Ave.
Redondo Beach, CA 90277

1 Dr. Gershon Weltman
Perceptronics, Inc.
6271 Variel Ave.
Woodland Hills, CA 91367

1 Dr. Keigh T. Wescourt
Information Sciences Dept.
The Rand Corporation
1700 Main St.
Santa Monica, CA 90406

1 Dr. Susan E. Whitely
Psychology Dept.
University of Kansas
Lawrence, Kansas 66044

1 Dr. Christopher Wickens
Dept. of Psychology
University of Illinois
Champaign, IL 61820

1 Dr. J. Arthur Woodward
Department of Psychology
University of California
Los Angeles, CA 90024

CHIP Technical Report List

1. David M. Green and William J. McGill. On the equivalence of detection probabilities and well known statistical quantities. October, 1969.
2. Donald A. Norman. Comments on the information structure of memory. October, 1969.
3. Norman H. Anderson. Functional measurement and psychophysical judgment. October, 1969.
4. James C. Shanteau. An additive decision-making model for sequential estimation and inference judgments. October, 1969.
5. Norman H. Anderson. Averaging model applied to the size-weight illusion. October, 1969.
6. Norman H. Anderson and James C. Shanteau. Information integration in risky decision making. November, 1969.
7. George Mandler, Richard H. Meltzer, Zena Pearlstone. The structure of recognition. Effects of list tags and of acoustic and semantic confusion. November, 1969.
8. Dominic W. Massaro. Perceptual processes and forgetting in memory tasks. January, 1970.
9. Daniel Graboi. Searching for targets: The effects of specific practice. February, 1970.
10. James H. Patterson and David M. Green. Discrimination of transient signals having identical energy spectra. February, 1970.
11. Donald A. Norman. Remembrance of things past. June, 1970.
12. Norman H. Anderson. Integration theory and attitude change. August, 1970.
13. A.D. Baddeley and J.R. Ecob. Reaction time and short-term memory: A trace strength alternative to the high-speed exhaustive scanning hypothesis. November, 1970.
14. A.D. Baddeley. Retrieval rules and semantic coding in short-term memory. December, 1970.
15. Roy D. Patterson. Residue pitch as a function of the number and relative phase of the component sinusoids. March, 1971.
16. George Mandler and Marilyn A. Borges. Effects of list differentiation, category membership and prior recall on recognition. May, 1971.
17. David E. Rumelhart, Peter H. Lindsay, and Donald A. Norman. A process model for long-term memory. May, 1971.
18. David E. Rumelhart and Adele A. Abrahamson. Toward a theory of analogical reasoning. July, 1971.
19. Martin F. Kaplan. How response dispositions integrate with stimulus information. August, 1971.
20. Martin F. Kaplan and Norman H. Anderson. Comparison of information integration and reinforcement models for interpersonal attraction. August, 1971.
21. David M. Green and R. Duncan Luce. Speed-accuracy tradeoff in auditory detection. September, 1971.
22. David E. Rumelhart. A multicomponent theory of confusion among briefly exposed alphabetic characters. November, 1971.
23. Norman H. Anderson and Arthur J. Farkas. New light on order effects in attitude change. March, 1972.
24. Norman H. Anderson. Information integration theory: A brief survey. April, 1972.
25. Donald A. Norman. Memory, knowledge, and the answering of questions. May, 1972.
26. David J. Weiss. Averaging: An empirical validity criterion for magnitude estimation.
Norman H. Anderson. Cross-task validation of functional measurement. June, 1972.
27. David E. Rumelhart and Patricia Siple. The process of recognizing tachistoscopically presented words. August, 1972.
28. Ebbe B. Ebbesen and Richard Bowers. The effects of proportion of risky to conservative arguments in a group discussion on risky shift. September, 1972.
29. Ebbe B. Ebbesen and Michael Haney. Flirting with death: Variables affecting risk taking on our nation's highways. September, 1972.
30. Norman H. Anderson. Algebraic models in perception. November, 1972.
31. Norman H. Anderson. Cognitive algebra: Information integration applied to social attribution. December, 1972.
32. Jean M. Mandler and Nancy L. Stein. Recall recognition of pictures by children as a function of organization and of distractor similarity. January, 1973.
33. David E. Rumelhart and Donald A. Norman. Active semantic networks as a model of human memory.
Marc Eisenstadt and Yaakov Kareev. Towards a model of human game playing. June, 1973.
34. George Mandler. Memory storage and retrieval: Some limits on the reach of attention and consciousness. July, 1973.
35. Kent L. Norman. A method of maximum likelihood estimation for stimulus integration theory. August, 1973.
36. Yaakov Kareev. A model of human game playing. August, 1973.
37. Donald A. Norman. Cognitive organization and learning. August, 1973.
38. The Center for Human Information Processing: A Five Year Report — 1968-73.

39. Larry D. Rosen and J. Edward Russo. Binary processing in multi-alternative choice. October, 1973.

40. Samuel Himmelfarb and Norman H. Anderson. Integration theory analysis of opinion attribution. December, 1973.

41. George Mandler. Consciousness: Respectable, useful, and probably necessary. March, 1974.

42. Norman H. Anderson. The problem of change-of-meaning. June, 1974.

43. Norman H. Anderson. Methods for studying information integration. June, 1974.

44. Norman H. Anderson. Basic experiments in person perception. June, 1974.

45. Norman H. Anderson. Algebraic models for information integration. June, 1974.

46. Ebbe B. Ebbesen and Vladimir J. Konečni. Cognitive algebra in legal decision making. September, 1974.

47. Norman H. Anderson. Equity judgments as information integration.
Arthur J. Farkas and Norman H. Anderson. Input summation and equity summation in multi-cue equity judgments. December, 1974.

48. George Mandler and Arthur Graesser II. Dimensional analysis and the locus of organization. January, 1975.

49. James L. McClelland. Preliminary letter identification in the perception of words and nonwords. April, 1975.

50. Donald A. Norman and Daniel G. Bobrow. On the role of active memory processes in perception and cognition. May, 1975.

51. J. Edward Russo. The value of unit price information. An information processing analysis of point-of-purchase decisions. June, 1975.

52. Elissa L. Newport. Motherese: The speech of mothers to young children. August, 1975.

53. Norman H. Anderson and Cheryl C. Graesser. An information integration analysis of attitude change in group discussion. September, 1975.

54. Lynn A. Cooper. Demonstration of a mental analog of an external rotation.
Lynn A. Cooper and Peter Podgorny. Mental transformations and visual comparison processes: Effects of complexity and similarity. October, 1975.

55. David E. Rumelhart and Andrew Ortony. The representation of knowledge in memory. January, 1976.

56. David E. Rumelhart. Toward an interactive model of reading. March, 1976.

57. Jean M. Mandler, Nancy S. Johnson, and Marsha DeForest. A structural analysis of stories and their recall: From "Once upon a time" to "Happily ever after". March, 1976.

58. David E. Rumelhart. Understanding and summarizing brief stories. April, 1976.

59. Lynn A. Cooper and Roger N. Shepard. Transformations on representations of objects in space. April, 1976.

60. George Mandler. Some attempts to study the rotation and reversal of integrated motor patterns. May, 1976.

61. Norman H. Anderson. Problems in using analysis of variance in balance theory. June, 1976.

62. Norman H. Anderson. Social perception and cognition. July, 1976.

63. David E. Rumelhart and Donald A. Norman. Accretion, tuning and restructuring: Three modes of learning. August, 1976.

64. George Mandler. Memory research reconsidered: A critical view of traditional methods and distinctions. September, 1976.

65. Norman H. Anderson and Michael D. Klitzner. Measurement of motivation.
Michael D. Klitzner and Norman H. Anderson. Motivation x expectancy x value: A functional measurement approach. November, 1976.

66. Vladimir J. Konečni. Some social, emotional, and cognitive determinants of aesthetic preference for melodies in complexity. December, 1976.

67. Hugh Mehan, Courtney B. Cazden, LaDonna Coles, Sue Fisher, Nick Maroules. The social organization of classroom lessons. December, 1976.

67a. Hugh Mehan, Courtney B. Cazden, LaDonna Coles, Sue Fisher, Nick Maroules. Appendices to the social organization of classroom lessons. December, 1976.

68. Norman H. Anderson. Integration theory applied to cognitive responses and attitudes. December, 1976.

69. Norman H. Anderson and Diane O. Cuneo. The height + width rule in children's judgments of quantity. June, 1977.
Norman H. Anderson and Clifford H. Butzin. Children's judgments of equity. June, 1977.

70. Donald R. Gentner and Donald A. Norman. The FLOW tutor: Schemas for tutoring. June, 1977.

71. George Mandler. Organization and repetition: An extension of organizational principles with special reference to rote learning. May, 1977.

72. Manuel Leon. Coordination of intent and consequence information in children's moral judgements. August, 1977.

73. Ted Supalla and Elissa L. Newport. How many seats in a chair? The derivation of nouns and verbs in American Sign Language. November, 1977.

74. Donald A. Norman and Daniel G. Bobrow. Descriptions: A basis for memory acquisition and retrieval. November, 1977.

75. Michael D. Williams. The process of retrieval from very long term memory. September, 1978.

76. Jean M. Mandler. Categorical and schematic organization in memory. October, 1978.

77. James L. McClelland. On time relations of mental processes: A framework for analyzing processes in cascade. October, 1978.

78. Jean M. Mandler and Marsha DeForest. Developmental invariance in story recall. November, 1978.

79. David E. Rumelhart. Schemata: The building blocks of cognition. December, 1978.

80. Nancy S. Johnson and Jean M. Mandler. A tale of two structures: Underlying and surface forms in stories. January, 1979.

81. David E. Rumelhart. Analogical processes and procedural representations. February, 1979.

82. Ross A. Bott. A study of complex learning: Theory and methodologies. March, 1979.

83. Laboratory of Comparative Human Cognition. Toward a unified approach to problems of culture and cognition. May, 1979.

84. George Mandler and Lawrence W. Barsalou. Steady state memory: What does the one-shot experiment assess? May, 1979.

85. Norman H. Anderson. Introduction to cognitive algebra. June, 1979.

86. Edited by Michael Cole, Edwin Hutchins, James Levin and Naomi Miyake. Naturalistic problem solving and microcomputers. Report of a Conference. June, 1979.

87. Donald A. Norman. Twelve issues for cognitive science. October, 1979.

88. Donald A. Norman. Slips of the mind and an outline for a theory of action. November, 1979.

89. The Center for Human Information Processing: A Description and a Five-Year Report (1974-1979). November, 1979.

90. Michael Cole and Peg Griffin. Cultural amplifiers reconsidered. December, 1979.

91. James L. McClelland and David E. Rumelhart. An interactive activation model of the effect of context in perception. Part I. April, 1980.

92. James L. McClelland and J.K. O'Regan. The role of expectations in the use of peripheral visual information in reading. February, 1980.

93. Edwin Hutchins. Conceptual structures of Caroline Island navigation. May, 1980.

94. Friedrich Wilkening and Norman H. Anderson. Comparison of two rule assessment methodologies for studying cognitive development. June, 1980.

95. David E. Rumelhart and James L. McClelland. An interactive activation model of the effect of context in perception. Part II. August, 1980.

96. Jean M. Mandler. Structural invariants in development. September, 1980.

97. David E. Rumelhart and Donald A. Norman. Analogical processes in learning. October, 1980.

98. James A. Levin and Yaakov Kareev. Personal computers and education: The challenge to schools. November, 1980.

99. Donald A. Norman and Tim Shallice. Attention to action: Willed and automatic control of behavior. December, 1980.

100. David E. Rumelhart. Understanding understanding. January, 1981.

101. George Mandler. The structure of value: Accounting for taste. May, 1981.

102. David E. Rumelhart and Donald A. Norman. Simulating a skilled typist: A study of skilled cognitive-motor performance. May, 1981.

103. Jean M. Mandler. Representation. June, 1981.

104. Donald R. Gentner. Skilled finger movements in typing. July, 1981.

105. Edwin L. Hutchins and James A. Levin. Point of view in problem solving. August, 1981.

106. Michael Cole. Society, mind and development. September, 1981.
Michael Cole. The zone of proximal development: Where culture and cognition create each other. September, 1981.

107. Laboratory of Comparative Human Cognition. Culture and cognitive development. November, 1981.

108. Donald R. Gentner. Evidence against a central control model of timing in typing. December, 1981.

109. Robert M. Boynton and Allen L. Nagy. The La Jolla analytic colorimeter: Optics, calibrations, procedures, and control experiments. December, 1981.

110. Jonathan T. Grudin and Serge Larochelle. Digraph frequency effects in skilled typing. February, 1982.

111. Donald R. Gentner, Jonathan Grudin, Serge Larochelle, Donald A. Norman, and David E. Rumelhart. Studies of typing from the LNR Research Group. May, 1982.

112. Donald A. Norman. Five papers on human-machine interaction. May, 1982.

Note: Requests for CHIP reports should be addressed to the author. Reports are also available through the Library Loan Service of the University of California, San Diego, La Jolla, California 92093.